

general_ledger

Generated by Doxygen 1.8.1.2

Sun Jun 8 2014 14:28:37

Contents

1	General Ledger.	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Data Structure Documentation	9
4.1	ds_list Struct Reference	9
4.1.1	Detailed Description	9
4.1.2	Field Documentation	9
4.1.2.1	current	10
4.1.2.2	data_destructor	10
4.1.2.3	free_on_delete	10
4.1.2.4	head	10
4.1.2.5	length	10
4.1.2.6	tail	10
4.2	ds_list_element Struct Reference	10
4.2.1	Detailed Description	10
4.2.2	Field Documentation	11
4.2.2.1	data	11
4.2.2.2	next	11
4.2.2.3	previous	11
4.3	ds_map Struct Reference	11
4.3.1	Detailed Description	11
4.3.2	Field Documentation	12
4.3.2.1	hash_size	12
4.3.2.2	lists	12
4.4	ds_map_str Struct Reference	12
4.4.1	Detailed Description	12
4.4.2	Field Documentation	12

4.4.2.1	hash_size	13
4.4.2.2	lists	13
4.5	ds_record Struct Reference	13
4.5.1	Detailed Description	13
4.5.2	Field Documentation	13
4.5.2.1	fields	13
4.6	ds_recordset Struct Reference	14
4.6.1	Detailed Description	14
4.6.2	Field Documentation	14
4.6.2.1	field_lengths	14
4.6.2.2	headers	14
4.6.2.3	num_fields	14
4.6.2.4	records	15
4.6.2.5	types	15
4.7	ds_str Struct Reference	15
4.7.1	Detailed Description	15
4.7.2	Field Documentation	15
4.7.2.1	capacity	15
4.7.2.2	data	15
4.7.2.3	length	15
4.8	ds_vector Struct Reference	15
4.8.1	Detailed Description	16
4.8.2	Field Documentation	16
4.8.2.1	current	16
4.8.2.2	data	16
4.8.2.3	data_destructor	16
4.8.2.4	free_on_delete	16
4.8.2.5	size	16
4.9	kv_pair_node Struct Reference	16
4.9.1	Detailed Description	17
4.9.2	Field Documentation	17
4.9.2.1	key	17
4.9.2.2	key	17
4.9.2.3	next	17
4.9.2.4	value	17
4.9.2.5	value	17
4.10	params Struct Reference	17
4.10.1	Detailed Description	18
4.10.2	Field Documentation	18
4.10.2.1	database	18

4.10.2.2	hostname	18
4.10.2.3	password	18
4.10.2.4	username	18
5	File Documentation	19
5.1	lib/database/database.h File Reference	19
5.1.1	Detailed Description	20
5.2	lib/database/db_connection.h File Reference	20
5.2.1	Detailed Description	20
5.2.2	Function Documentation	21
5.2.2.1	db_connect	21
5.3	lib/database/db_entities.c File Reference	21
5.3.1	Detailed Description	22
5.3.2	Function Documentation	22
5.3.2.1	db_create_entities_table	22
5.3.2.2	db_drop_entities_table	22
5.3.2.3	db_list_entities_report	22
5.4	lib/database/db_entities.h File Reference	22
5.4.1	Detailed Description	23
5.4.2	Function Documentation	24
5.4.2.1	db_create_entities_table	24
5.4.2.2	db_drop_entities_table	24
5.4.2.3	db_list_entities_report	24
5.5	lib/database/db_internal.h File Reference	24
5.5.1	Detailed Description	25
5.6	lib/database/db_jelines.c File Reference	25
5.6.1	Detailed Description	25
5.6.2	Function Documentation	26
5.6.2.1	db_create_jelines_table	26
5.6.2.2	db_drop_jelines_table	26
5.6.2.3	db_list_jelines_report	26
5.7	lib/database/db_jelines.h File Reference	26
5.7.1	Detailed Description	27
5.7.2	Function Documentation	28
5.7.2.1	db_create_jelines_table	28
5.7.2.2	db_drop_jelines_table	28
5.7.2.3	db_list_jelines_report	28
5.8	lib/database/db_jes.c File Reference	28
5.8.1	Detailed Description	29
5.8.2	Function Documentation	29

5.8.2.1	db_create_jes_table	29
5.8.2.2	db_drop_jes_table	29
5.8.2.3	db_list_jes_report	29
5.9	lib/database/db_jes.h File Reference	29
5.9.1	Detailed Description	30
5.9.2	Function Documentation	31
5.9.2.1	db_create_jes_table	31
5.9.2.2	db_drop_jes_table	31
5.9.2.3	db_list_jes_report	31
5.10	lib/database/db_jesrcs.c File Reference	31
5.10.1	Detailed Description	32
5.10.2	Function Documentation	32
5.10.2.1	db_create_jesrcs_table	32
5.10.2.2	db_drop_jesrcs_table	32
5.10.2.3	db_list_jesrcs_report	32
5.11	lib/database/db_jesrcs.h File Reference	32
5.11.1	Detailed Description	33
5.11.2	Function Documentation	34
5.11.2.1	db_create_jesrcs_table	34
5.11.2.2	db_drop_jesrcs_table	34
5.11.2.3	db_list_jesrcs_report	34
5.12	lib/database/db_nomaccts.c File Reference	34
5.12.1	Detailed Description	35
5.12.2	Function Documentation	35
5.12.2.1	db_create_nomaccts_table	35
5.12.2.2	db_drop_nomaccts_table	35
5.12.2.3	db_list_nomaccts_report	35
5.13	lib/database/db_nomaccts.h File Reference	35
5.13.1	Detailed Description	36
5.13.2	Function Documentation	37
5.13.2.1	db_create_nomaccts_table	37
5.13.2.2	db_drop_nomaccts_table	37
5.13.2.3	db_list_nomaccts_report	37
5.14	lib/database/db_query.h File Reference	37
5.14.1	Detailed Description	38
5.14.2	Function Documentation	38
5.14.2.1	db_execute_query	38
5.15	lib/database/db_reporting.c File Reference	38
5.15.1	Detailed Description	39
5.15.2	Function Documentation	39

5.15.2.1	db_create_report_from_query	39
5.15.2.2	db_current_trial_balance_report	39
5.16	lib/database/db_reporting.h File Reference	39
5.16.1	Detailed Description	40
5.16.2	Function Documentation	40
5.16.2.1	db_create_recordset_from_query	40
5.16.2.2	db_create_report_from_query	40
5.16.2.3	db_current_trial_balance_report	40
5.17	lib/database/db_sampledata.c File Reference	41
5.17.1	Detailed Description	41
5.18	lib/database/db_sampledata.h File Reference	41
5.18.1	Detailed Description	42
5.19	lib/database/db_sql.h File Reference	42
5.19.1	Detailed Description	44
5.19.2	Function Documentation	44
5.19.2.1	db_create_entities_table_sql	44
5.19.2.2	db_create_jelines_table_sql	44
5.19.2.3	db_create_jes_table_sql	44
5.19.2.4	db_create_jesrcs_table_sql	44
5.19.2.5	db_create_nomaccts_table_sql	44
5.19.2.6	db_create_standingdata_table_sql	45
5.19.2.7	db_create_users_table_sql	45
5.19.2.8	db_current_trial_balance_report_sql	45
5.19.2.9	db_drop_entities_table_sql	45
5.19.2.10	db_drop_jelines_table_sql	45
5.19.2.11	db_drop_jes_table_sql	45
5.19.2.12	db_drop_jesrcs_table_sql	45
5.19.2.13	db_drop_nomaccts_table_sql	46
5.19.2.14	db_drop_standingdata_table_sql	46
5.19.2.15	db_drop_users_table_sql	46
5.19.2.16	db_list_entities_report_sql	46
5.19.2.17	db_list_jelines_report_sql	46
5.19.2.18	db_list_jes_report_sql	46
5.19.2.19	db_list_jesrcs_report_sql	46
5.19.2.20	db_list_nomaccts_report_sql	47
5.19.2.21	db_list_users_report_sql	47
5.19.2.22	db_show_standingdata_report_sql	47
5.20	lib/database/db_standingdata.c File Reference	47
5.20.1	Detailed Description	48
5.20.2	Function Documentation	48

5.20.2.1	db_create_standingdata_table	48
5.20.2.2	db_drop_standingdata_table	48
5.20.2.3	db_show_standingdata_report	48
5.21	lib/database/db_standingdata.h File Reference	48
5.21.1	Detailed Description	49
5.21.2	Function Documentation	50
5.21.2.1	db_create_standingdata_table	50
5.21.2.2	db_drop_standingdata_table	50
5.21.2.3	db_show_standingdata_report	50
5.22	lib/database/db_structure.c File Reference	50
5.22.1	Detailed Description	51
5.22.2	Function Documentation	51
5.22.2.1	db_create_database_structure	51
5.22.2.2	db_delete_database_structure	51
5.23	lib/database/db_structure.h File Reference	51
5.23.1	Detailed Description	52
5.23.2	Function Documentation	52
5.23.2.1	db_create_database_structure	52
5.23.2.2	db_delete_database_structure	52
5.24	lib/database/db_users.c File Reference	52
5.24.1	Detailed Description	53
5.24.2	Function Documentation	53
5.24.2.1	db_create_users_table	53
5.24.2.2	db_drop_users_table	54
5.24.2.3	db_list_users_report	54
5.25	lib/database/db_users.h File Reference	54
5.25.1	Detailed Description	55
5.25.2	Function Documentation	55
5.25.2.1	db_create_users_table	55
5.25.2.2	db_drop_users_table	55
5.25.2.3	db_list_users_report	55
5.26	lib/database/dummy/db_dummy_create_entities_table_sql.c File Reference	55
5.26.1	Detailed Description	56
5.26.2	Function Documentation	56
5.26.2.1	db_create_entities_table_sql	56
5.27	lib/database/dummy/db_dummy_create_users_table_sql.c File Reference	56
5.27.1	Detailed Description	56
5.27.2	Function Documentation	56
5.27.2.1	db_create_users_table_sql	56
5.28	lib/database/dummy/db_dummy_drop_entities_table_sql.c File Reference	57

5.28.1 Detailed Description	57
5.28.2 Function Documentation	57
5.28.2.1 db_drop_entities_table_sql	57
5.29 lib/database/dummy/db_dummy_drop_users_table_sql.c File Reference	57
5.29.1 Detailed Description	57
5.29.2 Function Documentation	58
5.29.2.1 db_drop_users_table_sql	58
5.30 lib/database/dummy/db_dummy_general.c File Reference	58
5.30.1 Detailed Description	59
5.30.2 Macro Definition Documentation	59
5.30.2.1 _XOPEN_SOURCE	59
5.30.3 Function Documentation	59
5.30.3.1 db_connect	59
5.30.3.2 db_create_recordset_from_query	59
5.30.3.3 db_execute_query	59
5.31 lib/database/dummy/db_dummy_list_entities_report_sql.c File Reference	60
5.31.1 Detailed Description	60
5.31.2 Function Documentation	60
5.31.2.1 db_list_entities_report_sql	60
5.32 lib/database/dummy/db_dummy_list_users_report_sql.c File Reference	60
5.32.1 Detailed Description	61
5.32.2 Function Documentation	61
5.32.2.1 db_list_users_report_sql	61
5.33 lib/database/mysql/db_mysql_create_entities_table_sql.c File Reference	61
5.33.1 Detailed Description	61
5.33.2 Function Documentation	61
5.33.2.1 db_create_entities_table_sql	61
5.34 lib/database/mysql/db_mysql_create_jelines_table_sql.c File Reference	62
5.34.1 Detailed Description	62
5.34.2 Function Documentation	62
5.34.2.1 db_create_jelines_table_sql	62
5.35 lib/database/mysql/db_mysql_create_jes_table_sql.c File Reference	62
5.35.1 Detailed Description	62
5.35.2 Function Documentation	63
5.35.2.1 db_create_jes_table_sql	63
5.36 lib/database/mysql/db_mysql_create_jesrcs_table_sql.c File Reference	63
5.36.1 Detailed Description	63
5.36.2 Function Documentation	63
5.36.2.1 db_create_jesrcs_table_sql	63
5.37 lib/database/mysql/db_mysql_create_nomaccts_table_sql.c File Reference	63

5.37.1 Detailed Description	64
5.37.2 Function Documentation	64
5.37.2.1 db_create_nomaccts_table_sql	64
5.38 lib/database/mysql/db_mysql_create_standingdata_table_sql.c File Reference	64
5.38.1 Detailed Description	64
5.38.2 Function Documentation	64
5.38.2.1 db_create_standingdata_table_sql	64
5.39 lib/database/mysql/db_mysql_create_users_table_sql.c File Reference	65
5.39.1 Detailed Description	65
5.39.2 Function Documentation	65
5.39.2.1 db_create_users_table_sql	65
5.40 lib/database/mysql/db_mysql_current_trial_balance_report_sql.c File Reference	65
5.40.1 Detailed Description	65
5.40.2 Function Documentation	66
5.40.2.1 db_current_trial_balance_report_sql	66
5.41 lib/database/mysql/db_mysql_drop_entities_table_sql.c File Reference	66
5.41.1 Detailed Description	66
5.41.2 Function Documentation	66
5.41.2.1 db_drop_entities_table_sql	66
5.42 lib/database/mysql/db_mysql_drop_jelines_table_sql.c File Reference	66
5.42.1 Detailed Description	67
5.42.2 Function Documentation	67
5.42.2.1 db_drop_jelines_table_sql	67
5.43 lib/database/mysql/db_mysql_drop_jes_table_sql.c File Reference	67
5.43.1 Detailed Description	67
5.43.2 Function Documentation	67
5.43.2.1 db_drop_jes_table_sql	67
5.44 lib/database/mysql/db_mysql_drop_jesrcs_table_sql.c File Reference	68
5.44.1 Detailed Description	68
5.44.2 Function Documentation	68
5.44.2.1 db_drop_jesrcs_table_sql	68
5.45 lib/database/mysql/db_mysql_drop_nomaccts_table_sql.c File Reference	68
5.45.1 Detailed Description	68
5.45.2 Function Documentation	69
5.45.2.1 db_drop_nomaccts_table_sql	69
5.46 lib/database/mysql/db_mysql_drop_standingdata_table_sql.c File Reference	69
5.46.1 Detailed Description	69
5.46.2 Function Documentation	69
5.46.2.1 db_drop_standingdata_table_sql	69
5.47 lib/database/mysql/db_mysql_drop_users_table_sql.c File Reference	69

5.47.1 Detailed Description	70
5.47.2 Function Documentation	70
5.47.2.1 db_drop_users_table_sql	70
5.48 lib/database/mysql/db_mysql_general.c File Reference	70
5.48.1 Detailed Description	71
5.48.2 Function Documentation	71
5.48.2.1 db_connect	71
5.48.2.2 db_create_recordset_from_query	71
5.48.2.3 db_execute_query	72
5.48.3 Variable Documentation	72
5.48.3.1 conn_mss	72
5.48.3.2 main_mss	72
5.49 lib/database/mysql/db_mysql_list_entities_report_sql.c File Reference	72
5.49.1 Detailed Description	72
5.49.2 Function Documentation	72
5.49.2.1 db_list_entities_report_sql	72
5.50 lib/database/mysql/db_mysql_list_jelines_report_sql.c File Reference	73
5.50.1 Detailed Description	73
5.50.2 Function Documentation	73
5.50.2.1 db_list_jelines_report_sql	73
5.51 lib/database/mysql/db_mysql_list_jes_report_sql.c File Reference	73
5.51.1 Detailed Description	73
5.51.2 Function Documentation	74
5.51.2.1 db_list_jes_report_sql	74
5.52 lib/database/mysql/db_mysql_list_jesrcs_report_sql.c File Reference	74
5.52.1 Detailed Description	74
5.52.2 Function Documentation	74
5.52.2.1 db_list_jesrcs_report_sql	74
5.53 lib/database/mysql/db_mysql_list_nomaccts_report_sql.c File Reference	74
5.53.1 Detailed Description	75
5.53.2 Function Documentation	75
5.53.2.1 db_list_nomaccts_report_sql	75
5.54 lib/database/mysql/db_mysql_list_users_report_sql.c File Reference	75
5.54.1 Detailed Description	75
5.54.2 Function Documentation	75
5.54.2.1 db_list_users_report_sql	75
5.55 lib/database/mysql/db_mysql_show_standingdata_report_sql.c File Reference	76
5.55.1 Detailed Description	76
5.55.2 Function Documentation	76
5.55.2.1 db_show_standingdata_report_sql	76

5.56	lib/datastruct/data_structures.h File Reference	76
5.56.1	Detailed Description	77
5.57	lib/datastruct/ds_fieldtypes.h File Reference	77
5.57.1	Detailed Description	78
5.57.2	Enumeration Type Documentation	78
5.57.2.1	ds_field_types	78
5.58	lib/datastruct/ds_list.c File Reference	78
5.58.1	Detailed Description	79
5.58.2	Function Documentation	80
5.58.2.1	ds_list_append	80
5.58.2.2	ds_list_create	80
5.58.2.3	ds_list_destroy	80
5.58.2.4	ds_list_destructor	80
5.58.2.5	ds_list_element	81
5.58.2.6	ds_list_get_next_data	81
5.58.2.7	ds_list_get_prev_data	81
5.58.2.8	ds_list_is_empty	81
5.58.2.9	ds_list_length	82
5.58.2.10	ds_list_remove_all	82
5.58.2.11	ds_list_remove_tail	82
5.58.2.12	ds_list_seek_end	82
5.58.2.13	ds_list_seek_start	82
5.59	lib/datastruct/ds_list.h File Reference	82
5.59.1	Detailed Description	84
5.59.2	Typedef Documentation	84
5.59.2.1	ds_list	84
5.59.3	Function Documentation	84
5.59.3.1	ds_list_append	84
5.59.3.2	ds_list_create	84
5.59.3.3	ds_list_destroy	85
5.59.3.4	ds_list_destructor	85
5.59.3.5	ds_list_element	85
5.59.3.6	ds_list_get_next_data	85
5.59.3.7	ds_list_get_prev_data	86
5.59.3.8	ds_list_is_empty	86
5.59.3.9	ds_list_length	86
5.59.3.10	ds_list_remove_all	86
5.59.3.11	ds_list_remove_tail	87
5.59.3.12	ds_list_seek_end	87
5.59.3.13	ds_list_seek_start	87

5.60 lib/datastruct/ds_map.c File Reference	87
5.60.1 Detailed Description	88
5.60.2 Function Documentation	88
5.60.2.1 ds_map_destroy	88
5.60.2.2 ds_map_get_value	88
5.60.2.3 ds_map_init	89
5.60.2.4 ds_map_insert	89
5.60.2.5 ds_map_print_all	89
5.61 lib/datastruct/ds_map.h File Reference	89
5.61.1 Detailed Description	90
5.61.2 Typedef Documentation	91
5.61.2.1 ds_map	91
5.61.3 Function Documentation	91
5.61.3.1 ds_map_destroy	91
5.61.3.2 ds_map_get_value	91
5.61.3.3 ds_map_init	91
5.61.3.4 ds_map_insert	91
5.61.3.5 ds_map_print_all	92
5.62 lib/datastruct/ds_map_str.c File Reference	92
5.62.1 Detailed Description	93
5.62.2 Function Documentation	93
5.62.2.1 ds_map_str_destroy	93
5.62.2.2 ds_map_str_get_value	93
5.62.2.3 ds_map_str_init	93
5.62.2.4 ds_map_str_insert	93
5.63 lib/datastruct/ds_map_str.h File Reference	94
5.63.1 Detailed Description	95
5.63.2 Typedef Documentation	95
5.63.2.1 ds_map_str	95
5.63.3 Function Documentation	95
5.63.3.1 ds_map_str_destroy	95
5.63.3.2 ds_map_str_get_value	95
5.63.3.3 ds_map_str_init	95
5.63.3.4 ds_map_str_insert	96
5.64 lib/datastruct/ds_record.c File Reference	96
5.64.1 Detailed Description	97
5.64.2 Function Documentation	97
5.64.2.1 ds_record_clear	97
5.64.2.2 ds_record_create	97
5.64.2.3 ds_record_destroy	98

5.64.2.4	ds_record_destructor	98
5.64.2.5	ds_record_get_field	98
5.64.2.6	ds_record_get_next_data	98
5.64.2.7	ds_record_make_delim_string	98
5.64.2.8	ds_record_make_values_string	99
5.64.2.9	ds_record_seek_start	99
5.64.2.10	ds_record_set_field	99
5.64.2.11	ds_record_size	99
5.64.2.12	ds_record_tokenize	100
5.65	lib/datastruct/ds_record.h File Reference	100
5.65.1	Detailed Description	101
5.65.2	Typedef Documentation	101
5.65.2.1	ds_record	101
5.65.3	Function Documentation	102
5.65.3.1	ds_record_clear	102
5.65.3.2	ds_record_create	102
5.65.3.3	ds_record_destroy	102
5.65.3.4	ds_record_destructor	102
5.65.3.5	ds_record_get_field	102
5.65.3.6	ds_record_get_next_data	103
5.65.3.7	ds_record_make_delim_string	103
5.65.3.8	ds_record_make_values_string	103
5.65.3.9	ds_record_seek_start	103
5.65.3.10	ds_record_set_field	103
5.65.3.11	ds_record_size	104
5.65.3.12	ds_record_tokenize	104
5.66	lib/datastruct/ds_recordset.c File Reference	104
5.66.1	Detailed Description	105
5.66.2	Function Documentation	106
5.66.2.1	ds_recordset_add_record	106
5.66.2.2	ds_recordset_create	106
5.66.2.3	ds_recordset_destroy	106
5.66.2.4	ds_recordset_get_next_insert_query	106
5.66.2.5	ds_recordset_get_text_report	107
5.66.2.6	ds_recordset_next_record	107
5.66.2.7	ds_recordset_num_fields	107
5.66.2.8	ds_recordset_num_records	107
5.66.2.9	ds_recordset_seek_start	108
5.66.2.10	ds_recordset_set_headers	108
5.66.2.11	ds_recordset_set_type	108

5.67	lib/datastruct/ds_recordset.h File Reference	108
5.67.1	Detailed Description	110
5.67.2	Typedef Documentation	110
5.67.2.1	ds_recordset	110
5.67.3	Function Documentation	110
5.67.3.1	ds_recordset_add_record	110
5.67.3.2	ds_recordset_create	110
5.67.3.3	ds_recordset_destroy	111
5.67.3.4	ds_recordset_get_next_insert_query	111
5.67.3.5	ds_recordset_get_text_report	111
5.67.3.6	ds_recordset_next_record	111
5.67.3.7	ds_recordset_num_fields	112
5.67.3.8	ds_recordset_num_records	112
5.67.3.9	ds_recordset_seek_start	112
5.67.3.10	ds_recordset_set_headers	112
5.67.3.11	ds_recordset_set_type	112
5.68	lib/datastruct/ds_str.c File Reference	113
5.68.1	Detailed Description	115
5.68.2	Function Documentation	115
5.68.2.1	ds_str_assign	115
5.68.2.2	ds_str_assign_cstr	115
5.68.2.3	ds_str_char_at_index	115
5.68.2.4	ds_str_clear	116
5.68.2.5	ds_str_compare	116
5.68.2.6	ds_str_compare_cstr	116
5.68.2.7	ds_str_concat	116
5.68.2.8	ds_str_concat_cstr	116
5.68.2.9	ds_str_create	117
5.68.2.10	ds_str_create_direct	117
5.68.2.11	ds_str_create_sprintf	117
5.68.2.12	ds_str_cstr	118
5.68.2.13	ds_str_decorate	118
5.68.2.14	ds_str_destroy	118
5.68.2.15	ds_str_destructor	118
5.68.2.16	ds_str_doubleval	118
5.68.2.17	ds_str_dup	119
5.68.2.18	ds_str_getline	119
5.68.2.19	ds_str_hash	119
5.68.2.20	ds_str_intval	119
5.68.2.21	ds_str_is_alnum	120

5.68.2.22 ds_str_is_empty	120
5.68.2.23 ds_str_length	120
5.68.2.24 ds_str_size_to_fit	120
5.68.2.25 ds_str_split	121
5.68.2.26 ds_str_strchr	121
5.68.2.27 ds_str_substr_left	121
5.68.2.28 ds_str_substr_right	121
5.68.2.29 ds_str_trim	122
5.68.2.30 ds_str_trim_leading	122
5.68.2.31 ds_str_trim_trailing	122
5.68.2.32 ds_str_trunc	122
5.69 lib/datastruct/ds_str.h File Reference	122
5.69.1 Detailed Description	125
5.69.2 Typedef Documentation	125
5.69.2.1 ds_str	125
5.69.3 Function Documentation	125
5.69.3.1 ds_str_assign	125
5.69.3.2 ds_str_assign_cstr	125
5.69.3.3 ds_str_char_at_index	125
5.69.3.4 ds_str_clear	126
5.69.3.5 ds_str_compare	126
5.69.3.6 ds_str_compare_cstr	126
5.69.3.7 ds_str_concat	126
5.69.3.8 ds_str_concat_cstr	127
5.69.3.9 ds_str_create	127
5.69.3.10 ds_str_create_direct	127
5.69.3.11 ds_str_create_sprintf	127
5.69.3.12 ds_str_cstr	128
5.69.3.13 ds_str_decorate	128
5.69.3.14 ds_str_destroy	128
5.69.3.15 ds_str_destructor	128
5.69.3.16 ds_str_doubleval	128
5.69.3.17 ds_str_dup	129
5.69.3.18 ds_str_getline	129
5.69.3.19 ds_str_hash	129
5.69.3.20 ds_str_intval	129
5.69.3.21 ds_str_is_alnum	130
5.69.3.22 ds_str_is_empty	130
5.69.3.23 ds_str_length	130
5.69.3.24 ds_str_size_to_fit	130

5.69.3.25 ds_str_split	131
5.69.3.26 ds_str_strchr	131
5.69.3.27 ds_str_substr_left	131
5.69.3.28 ds_str_substr_right	131
5.69.3.29 ds_str_trim	132
5.69.3.30 ds_str_trim_leading	132
5.69.3.31 ds_str_trim_trailing	132
5.69.3.32 ds_str_trunc	132
5.70 lib/datastruct/ds_vector.c File Reference	132
5.70.1 Detailed Description	133
5.70.2 Function Documentation	134
5.70.2.1 ds_vector_clear	134
5.70.2.2 ds_vector_create	134
5.70.2.3 ds_vector_destroy	134
5.70.2.4 ds_vector_destructor	134
5.70.2.5 ds_vector_element	135
5.70.2.6 ds_vector_get_next_data	135
5.70.2.7 ds_vector_seek_start	135
5.70.2.8 ds_vector_set	135
5.70.2.9 ds_vector_size	136
5.71 lib/datastruct/ds_vector.h File Reference	136
5.71.1 Detailed Description	137
5.71.2 Typedef Documentation	137
5.71.2.1 ds_vector	137
5.71.3 Function Documentation	137
5.71.3.1 ds_vector_clear	137
5.71.3.2 ds_vector_create	137
5.71.3.3 ds_vector_destroy	138
5.71.3.4 ds_vector_destructor	138
5.71.3.5 ds_vector_element	138
5.71.3.6 ds_vector_get_next_data	138
5.71.3.7 ds_vector_seek_start	139
5.71.3.8 ds_vector_set	139
5.71.3.9 ds_vector_size	139
5.72 lib/file_ops/config_file_read.c File Reference	139
5.72.1 Detailed Description	140
5.72.2 Macro Definition Documentation	141
5.72.2.1 CONFIG_MAP_SIZE	141
5.72.2.2 MAX_BUFFER_SIZE	141
5.72.3 Function Documentation	141

5.72.3.1	config_file_read	141
5.72.3.2	config_free	141
5.72.3.3	config_init	141
5.72.3.4	config_value_get	141
5.72.3.5	config_value_get_cstr	142
5.72.3.6	config_value_set	142
5.73	lib/file_ops/config_file_read.h File Reference	142
5.73.1	Detailed Description	144
5.73.2	Macro Definition Documentation	144
5.73.2.1	CONFIG_FILE_MALFORMED_FILE	144
5.73.2.2	CONFIG_FILE_NO_FILE	144
5.73.2.3	CONFIG_FILE_OK	144
5.73.3	Function Documentation	144
5.73.3.1	config_file_read	144
5.73.3.2	config_free	144
5.73.3.3	config_init	145
5.73.3.4	config_value_get	145
5.73.3.5	config_value_get_cstr	145
5.73.3.6	config_value_set	145
5.74	lib/file_ops/delim_file_read.c File Reference	145
5.74.1	Detailed Description	146
5.74.2	Macro Definition Documentation	147
5.74.2.1	MAX_LINE_SIZE	147
5.74.3	Function Documentation	147
5.74.3.1	delim_file_read	147
5.75	lib/file_ops/delim_file_read.h File Reference	147
5.75.1	Detailed Description	148
5.75.2	Function Documentation	148
5.75.2.1	delim_file_read	148
5.76	lib/file_ops/file_ops.h File Reference	148
5.76.1	Detailed Description	149
5.77	lib/gl_general/gl_config.c File Reference	149
5.77.1	Detailed Description	150
5.77.2	Function Documentation	150
5.77.2.1	get_configuration	150
5.77.2.2	params_free	151
5.77.2.3	params_init	151
5.78	lib/gl_general/gl_config.h File Reference	151
5.78.1	Detailed Description	152
5.78.2	Function Documentation	152

5.78.2.1	get_configuration	152
5.78.2.2	params_free	152
5.78.2.3	params_init	153
5.79	lib/gl_general/gl_errors.c File Reference	153
5.79.1	Detailed Description	153
5.79.2	Function Documentation	154
5.79.2.1	gl_error_quit	154
5.80	lib/gl_general/gl_errors.h File Reference	154
5.80.1	Detailed Description	154
5.80.2	Function Documentation	154
5.80.2.1	gl_error_quit	154
5.81	lib/gl_general/gl_general.h File Reference	155
5.81.1	Detailed Description	155
5.82	lib/gl_general/gl_logging.c File Reference	156
5.82.1	Detailed Description	156
5.82.2	Function Documentation	156
5.82.2.1	gl_log_msg	156
5.82.2.2	gl_set_logging	157
5.83	lib/gl_general/gl_logging.h File Reference	157
5.83.1	Detailed Description	157
5.83.2	Function Documentation	158
5.83.2.1	gl_log_msg	158
5.83.2.2	gl_set_logging	158
5.84	lib/gl_general/gl_login.c File Reference	158
5.84.1	Detailed Description	159
5.84.2	Function Documentation	159
5.84.2.1	login	159
5.85	lib/gl_general/gl_login.h File Reference	160
5.85.1	Detailed Description	160
5.85.2	Function Documentation	160
5.85.2.1	login	161
5.86	progs/gl_db/gl_db_config.c File Reference	161
5.86.1	Detailed Description	161
5.86.2	Macro Definition Documentation	162
5.86.2.1	_XOPEN_SOURCE	162
5.86.3	Function Documentation	162
5.86.3.1	get_cmdline_options	162
5.87	progs/gl_db/gl_db_config.h File Reference	162
5.87.1	Detailed Description	163
5.87.2	Function Documentation	164

5.87.2.1	get_cmdline_options	164
5.88	progs/gl_db/gl_db_main.c File Reference	164
5.88.1	Detailed Description	165
5.88.2	Function Documentation	165
5.88.2.1	main	165
5.88.2.2	print_help_message	165
5.88.2.3	print_usage_message	165
5.88.2.4	print_version_message	165
5.89	progs/gl_reports/gl_reports_config.c File Reference	166
5.89.1	Detailed Description	166
5.89.2	Macro Definition Documentation	167
5.89.2.1	_XOPEN_SOURCE	167
5.89.3	Function Documentation	167
5.89.3.1	get_cmdline_options	167
5.90	progs/gl_reports/gl_reports_config.h File Reference	167
5.90.1	Detailed Description	168
5.90.2	Function Documentation	168
5.90.2.1	get_cmdline_options	168

Chapter 1

General Ledger.

General Ledger will be a fully-featured, multi-user, open-source general ledger system. The project is in the early stages of development.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

ds_list	9
ds_list_element	10
ds_map	11
ds_map_str	12
ds_record	13
ds_recordset	14
ds_str	15
ds_vector	15
kv_pair_node	16
params	17

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

lib/database/ database.h	
User interface to database functionality	19
lib/database/ db_connection.h	
Interface to database connection functionality	20
lib/database/ db_entities.c	
Implementation of entities functionality	21
lib/database/ db_entities.h	
Interface to entities functionality	22
lib/database/ db_internal.h	
Internal library interface to database functionality	24
lib/database/ db_jelines.c	
Implementation of journal entries functionality	25
lib/database/ db_jelines.h	
Interface to journal entry lines functionality	26
lib/database/ db_jes.c	
Implementation of journal entries functionality	28
lib/database/ db_jes.h	
Interface to journal entries functionality	29
lib/database/ db_jesrcs.c	
Implementation of journal entry sources functionality	31
lib/database/ db_jesrcs.h	
Interface to journal entry sources functionality	32
lib/database/ db_nomaccts.c	
Implementation of nominal accounts functionality	34
lib/database/ db_nomaccts.h	
Interface to nominal accounts functionality	35
lib/database/ db_query.h	
Interface to database query functionality	37
lib/database/ db_reporting.c	
Implementation of database reporting functionality	38
lib/database/ db_reporting.h	
Interface to database reporting functionality	39
lib/database/ db_sampledata.c	
Implementation of database sample data functionality	41
lib/database/ db_sampledata.h	
Interface to database sample data functionality	41
lib/database/ db_sql.h	
Interface to database specific SQL strings	42

lib/database/db_standingdata.c	
Implementation of standing data functionality	47
lib/database/db_standingdata.h	
Interface to journal entries functionality	48
lib/database/db_structure.c	
Implementation of database structure functionality	50
lib/database/db_structure.h	
Interface to database structure functionality	51
lib/database/db_users.c	
Implementation of users functionality	52
lib/database/db_users.h	
Interface to users functionality	54
lib/database/dummy/db_dummy_create_entities_table_sql.c	
Returns dummy SQL query to create entities table	55
lib/database/dummy/db_dummy_create_users_table_sql.c	
Returns dummy SQL query to create users table	56
lib/database/dummy/db_dummy_drop_entities_table_sql.c	
Returns dummy SQL query to drop entities table	57
lib/database/dummy/db_dummy_drop_users_table_sql.c	
Returns dummy SQL query to drop users table	57
lib/database/dummy/db_dummy_general.c	
Implementation of dummy database functionality	58
lib/database/dummy/db_dummy_list_entities_report_sql.c	
Returns dummy SQL query to create list entities report	60
lib/database/dummy/db_dummy_list_users_report_sql.c	
Returns dummy SQL query to create list users report	60
lib/database/mysql/db_mysql_create_entities_table_sql.c	
Returns MYSQL SQL query to create entities table	61
lib/database/mysql/db_mysql_create_jelines_table_sql.c	
Returns MYSQL SQL query to create journal entry lines table	62
lib/database/mysql/db_mysql_create_jes_table_sql.c	
Returns MYSQL SQL query to create journal entries table	62
lib/database/mysql/db_mysql_create_jesrcs_table_sql.c	
Returns MYSQL SQL query to create JE sources table	63
lib/database/mysql/db_mysql_create_nomaccts_table_sql.c	
Returns MYSQL SQL query to create nominal accounts table	63
lib/database/mysql/db_mysql_create_standingdata_table_sql.c	
Returns MYSQL SQL query to create standing data table	64
lib/database/mysql/db_mysql_create_users_table_sql.c	
Returns MYSQL SQL query to create users table	65
lib/database/mysql/db_mysql_current_trial_balance_report_sql.c	
Returns MYSQL SQL query to create current TB report	65
lib/database/mysql/db_mysql_drop_entities_table_sql.c	
Returns MYSQL SQL query to drop entities table	66
lib/database/mysql/db_mysql_drop_jelines_table_sql.c	
Returns MYSQL SQL query to drop journal entry lines table	66
lib/database/mysql/db_mysql_drop_jes_table_sql.c	
Returns MYSQL SQL query to drop entities table	67
lib/database/mysql/db_mysql_drop_jesrcs_table_sql.c	
Returns MYSQL SQL query to drop JE sources table	68
lib/database/mysql/db_mysql_drop_nomaccts_table_sql.c	
Returns MYSQL SQL query to drop nominal accounts table	68
lib/database/mysql/db_mysql_drop_standingdata_table_sql.c	
Returns MYSQL SQL query to drop standing data table	69
lib/database/mysql/db_mysql_drop_users_table_sql.c	
Returns MYSQL SQL query to drop users table	69
lib/database/mysql/db_mysql_general.c	
Implementation of MYSQL database functionality	70

lib/database/mysql/db_mysql_list_entities_report_sql.c	
Returns MYSQL SQL query to create list entities report	72
lib/database/mysql/db_mysql_list_jelines_report_sql.c	
Returns MYSQL SQL query to create JE lines report	73
lib/database/mysql/db_mysql_list_jes_report_sql.c	
Returns MYSQL SQL query to create journal entries report	73
lib/database/mysql/db_mysql_list_jesrcs_report_sql.c	
Returns MYSQL SQL query to create JE sources report	74
lib/database/mysql/db_mysql_list_nomaccts_report_sql.c	
Returns MYSQL SQL query to create list nominal accounts report	74
lib/database/mysql/db_mysql_list_users_report_sql.c	
Returns MYSQL SQL query to create list users report	75
lib/database/mysql/db_mysql_show_standingdata_report_sql.c	
Returns MYSQL SQL query to create show standing data report	76
lib/datastruct/data_structures.h	
Interface to data structures	76
lib/datastruct/ds_fieldtypes.h	
Record field types enumeration	77
lib/datastruct/ds_list.c	
Implementation of generic doubly-linked list data structure	78
lib/datastruct/ds_list.h	
Interface to generic doubly-linked list data structure	82
lib/datastruct/ds_map.c	
Implementation of string-string hash map data structure	87
lib/datastruct/ds_map.h	
Interface to string-string hash map data structure	89
lib/datastruct/ds_map_str.c	
Implementation of string-string hash map data structure	92
lib/datastruct/ds_map_str.h	
Interface to string-string hash map data structure	94
lib/datastruct/ds_record.c	
Implementation of record database structure	96
lib/datastruct/ds_record.h	
Interface to record data structure	100
lib/datastruct/ds_recordset.c	
Implementation of query result set structure	104
lib/datastruct/ds_recordset.h	
Interface to record set structure	108
lib/datastruct/ds_str.c	
Implementation of string data structure	113
lib/datastruct/ds_str.h	
Interface to string data structure	122
lib/datastruct/ds_vector.c	
Implementation of generic doubly-linked vector data structure	132
lib/datastruct/ds_vector.h	
Interface to generic doubly-linked vector data structure	136
lib/file_ops/config_file_read.c	
Implementation of configuration file reading functionality	139
lib/file_ops/config_file_read.h	
Interface to configuration file reading functionality	142
lib/file_ops/delim_file_read.c	
Implementation of delimited file reading functionality	145
lib/file_ops/delim_file_read.h	
Interface to delimited file reading functionality	147
lib/file_ops/file_ops.h	
User interface to file operations functionality	148
lib/gl_general/gl_config.c	
Implementation of configuration functionality	149

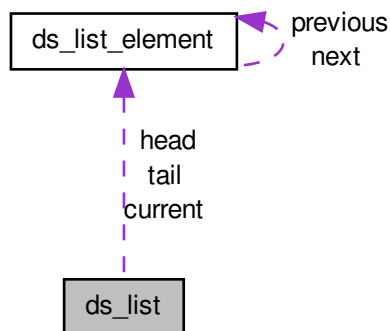
lib/gl_general/ gl_config.h	
Interface to configuration functionality	151
lib/gl_general/ gl_errors.c	
Implementation of error functionality	153
lib/gl_general/ gl_errors.h	
Interface to error functionality	154
lib/gl_general/ gl_general.h	
User interface to logging and error functionality	155
lib/gl_general/ gl_logging.c	
Implementation of logging functionality	156
lib/gl_general/ gl_logging.h	
Interface to logging functionality	157
lib/gl_general/ gl_login.c	
Implementation of login functionality	158
lib/gl_general/ gl_login.h	
Interface to login functionality	160
progs/gl_db/ gl_db_config.c	
Implementation of GL DB program configuration functionality	161
progs/gl_db/ gl_db_config.h	
Interface to GL DB program configuration functionality	162
progs/gl_db/ gl_db_main.c	
Main function for GL database program	164
progs/gl_reports/ gl_reports_config.c	
Implementation of GL reports program configuration functionality	166
progs/gl_reports/ gl_reports_config.h	
Interface to GL reports program configuration functionality	167

Chapter 4

Data Structure Documentation

4.1 ds_list Struct Reference

Collaboration diagram for ds_list:



Data Fields

- `size_t` `length`
- `bool` `free_on_delete`
- `struct ds_list_element *` `head`
- `struct ds_list_element *` `tail`
- `struct ds_list_element *` `current`
- `void(* data_destructor)(void *)`

4.1.1 Detailed Description

List data structure

4.1.2 Field Documentation

4.1.2.1 struct `ds_list_element*` `ds_list::current`

Pointer to current element

4.1.2.2 void(* `ds_list::data_destructor`)(void *)

Data destructor function

4.1.2.3 bool `ds_list::free_on_delete`

'Free on delete' flag

4.1.2.4 struct `ds_list_element*` `ds_list::head`

Pointer to head element

4.1.2.5 size_t `ds_list::length`

Length of list

4.1.2.6 struct `ds_list_element*` `ds_list::tail`

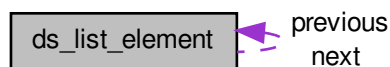
Pointer to tail element

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_list.c](#)

4.2 ds_list_element Struct Reference

Collaboration diagram for `ds_list_element`:



Data Fields

- void * [data](#)
- struct [ds_list_element](#) * [previous](#)
- struct [ds_list_element](#) * [next](#)

4.2.1 Detailed Description

List element data structure

4.2.2 Field Documentation

4.2.2.1 void* ds_list_element::data

Pointer to data

4.2.2.2 struct ds_list_element* ds_list_element::next

Pointer to next element

4.2.2.3 struct ds_list_element* ds_list_element::previous

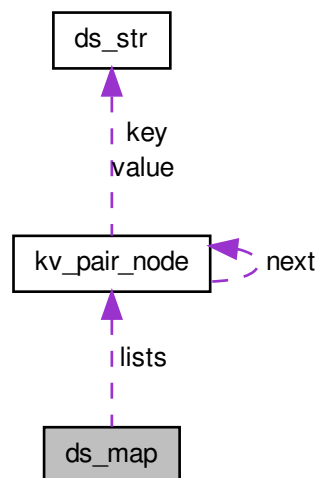
Pointer to previous element

The documentation for this struct was generated from the following file:

- lib/datastruct/[ds_list.c](#)

4.3 ds_map Struct Reference

Collaboration diagram for ds_map:



Data Fields

- struct [kv_pair_node](#) ** `lists`
- size_t [hash_size](#)

4.3.1 Detailed Description

Structure to hold a hash map

4.3.2 Field Documentation

4.3.2.1 `size_t ds_map::hash_size`

Size of array of lists

4.3.2.2 `struct kv_pair_node** ds_map::lists`

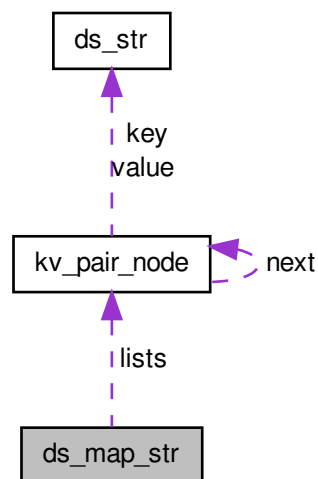
Pointer to array of lists

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_map.c](#)

4.4 `ds_map_str` Struct Reference

Collaboration diagram for `ds_map_str`:



Data Fields

- struct [kv_pair_node](#) ** `lists`
- `size_t` [hash_size](#)

4.4.1 Detailed Description

Structure to hold a hash map

4.4.2 Field Documentation

4.4.2.1 `size_t ds_map_str::hash_size`

Size of array of lists

4.4.2.2 `struct kv_pair_node** ds_map_str::lists`

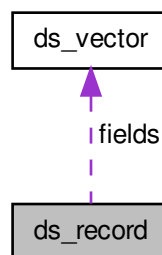
Pointer to array of lists

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_map_str.c](#)

4.5 ds_record Struct Reference

Collaboration diagram for ds_record:



Data Fields

- struct [ds_vector](#) * `fields`

4.5.1 Detailed Description

Vector data structure

4.5.2 Field Documentation

4.5.2.1 `struct ds_vector* ds_record::fields`

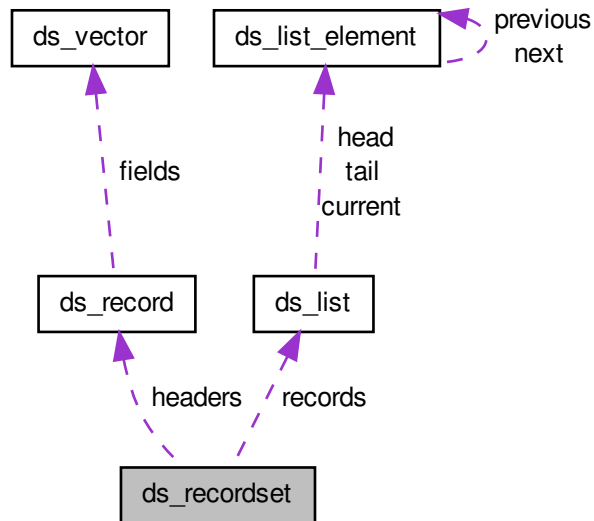
Vector of fields

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_record.c](#)

4.6 ds_recordset Struct Reference

Collaboration diagram for ds_recordset:



Data Fields

- `size_t num_fields`
- `size_t * field_lengths`
- `ds_record headers`
- `ds_list records`
- `enum ds_field_types * types`

4.6.1 Detailed Description

Result set structure

4.6.2 Field Documentation

4.6.2.1 `size_t* ds_recordset::field_lengths`

Lengths of the longest fields

4.6.2.2 `ds_record ds_recordset::headers`

A list of field headers

4.6.2.3 `size_t ds_recordset::num_fields`

The number of fields in a record

4.6.2.4 ds_list ds_recordset::records

A list of records

4.6.2.5 enum ds_field_types* ds_recordset::types

Types of records

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_recordset.c](#)

4.7 ds_str Struct Reference

Data Fields

- `char *` [data](#)
- `size_t` [length](#)
- `size_t` [capacity](#)

4.7.1 Detailed Description

Structure to contain string

4.7.2 Field Documentation

4.7.2.1 size_t ds_str::capacity

The size of the `data` buffer

4.7.2.2 char* ds_str::data

The data in C-style string format

4.7.2.3 size_t ds_str::length

The length of the string

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_str.c](#)

4.8 ds_vector Struct Reference

Data Fields

- `size_t` [size](#)
- `size_t` [current](#)
- `bool` [free_on_delete](#)
- `void **` [data](#)
- `void(*` [data_destructor](#) `)(void *)`

4.8.1 Detailed Description

Vector data structure

4.8.2 Field Documentation

4.8.2.1 `size_t ds_vector::current`

Current position

4.8.2.2 `void** ds_vector::data`

Data array

4.8.2.3 `void(* ds_vector::data_destructor)(void *)`

Data destructor function

4.8.2.4 `bool ds_vector::free_on_delete`

'Free on delete' flag

4.8.2.5 `size_t ds_vector::size`

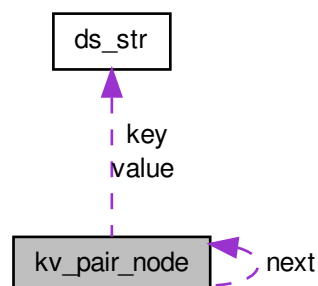
Size of vector

The documentation for this struct was generated from the following file:

- [lib/datastruct/ds_vector.c](#)

4.9 kv_pair_node Struct Reference

Collaboration diagram for kv_pair_node:



Data Fields

- char * [key](#)
- char * [value](#)
- struct [kv_pair_node](#) * [next](#)
- [ds_str](#) [key](#)
- [ds_str](#) [value](#)

4.9.1 Detailed Description

Structure to hold a key-value pair node

4.9.2 Field Documentation

4.9.2.1 [ds_str](#) [kv_pair_node::key](#)

A pointer to the key

4.9.2.2 [char*](#) [kv_pair_node::key](#)

A pointer to the key

4.9.2.3 [struct kv_pair_node *](#) [kv_pair_node::next](#)

A pointer to the next node

4.9.2.4 [ds_str](#) [kv_pair_node::value](#)

A pointer to the value

4.9.2.5 [char*](#) [kv_pair_node::value](#)

A pointer to the value

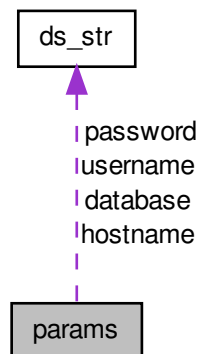
The documentation for this struct was generated from the following files:

- [lib/datastruct/ds_map.c](#)
- [lib/datastruct/ds_map_str.c](#)

4.10 params Struct Reference

```
#include <gl_config.h>
```

Collaboration diagram for params:



Data Fields

- [ds_str hostname](#)
- [ds_str database](#)
- [ds_str username](#)
- [ds_str password](#)

4.10.1 Detailed Description

Structure to hold database login parameters

4.10.2 Field Documentation

4.10.2.1 `ds_str params::database`

Database name

4.10.2.2 `ds_str params::hostname`

Database hostname

4.10.2.3 `ds_str params::password`

Password for database access

4.10.2.4 `ds_str params::username`

Username for database access

The documentation for this struct was generated from the following file:

- [lib/gl_general/gl_config.h](#)

Chapter 5

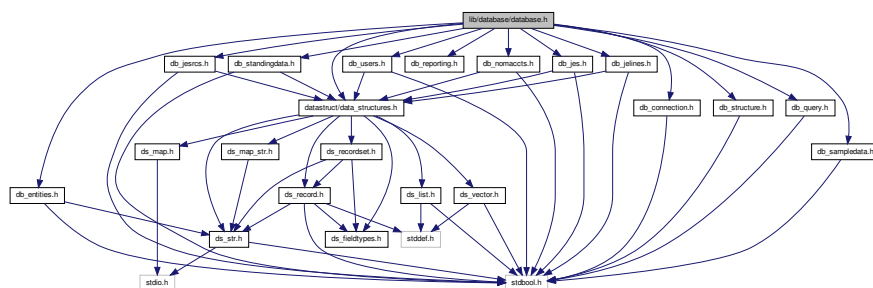
File Documentation

5.1 lib/database/database.h File Reference

User interface to database functionality.

```
#include "datastruct/data_structures.h"
#include "db_connection.h"
#include "db_structure.h"
#include "db_query.h"
#include "db_sampledata.h"
#include "db_reporting.h"
#include "db_users.h"
#include "db_entities.h"
#include "db_nomaccts.h"
#include "db_jes.h"
#include "db_jelines.h"
#include "db_jesrcs.h"
#include "db_standingdata.h"
```

Include dependency graph for database.h:



This graph shows which files directly or indirectly include this file:



5.1.1 Detailed Description

User interface to database functionality.

Author

Paul Griffiths

Copyright

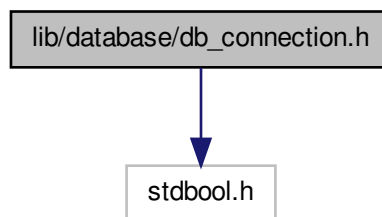
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.2 lib/database/db_connection.h File Reference

Interface to database connection functionality.

```
#include <stdbool.h>
```

Include dependency graph for db_connection.h:



This graph shows which files directly or indirectly include this file:



Functions

- bool **db_connect** (const char *host, const char *database, const char *username, const char *password)
Connects to a database.
- void **db_close** (void)
Disconnects from a database.

5.2.1 Detailed Description

Interface to database connection functionality. Function implementations are provided by the individual database components.

Creates a report listing all entities.

5.3.1 Detailed Description

Implementation of entities functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.3.2 Function Documentation

5.3.2.1 `bool db_create_entities_table (void)`

Creates the entities table in the database.

Returns

`true` on success, `false` on failure.

5.3.2.2 `bool db_drop_entities_table (void)`

Drops the entities table in the database.

Returns

`true` on success, `false` on failure.

5.3.2.3 `ds_str db_list_entities_report (void)`

Creates a report listing all entities.

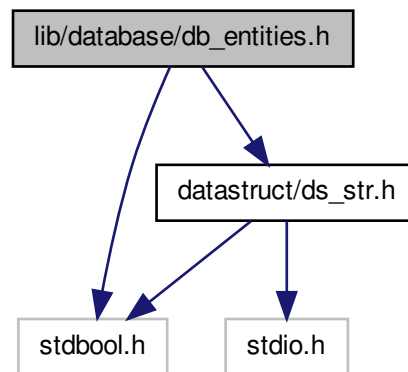
Returns

A `ds_str` containing the report.

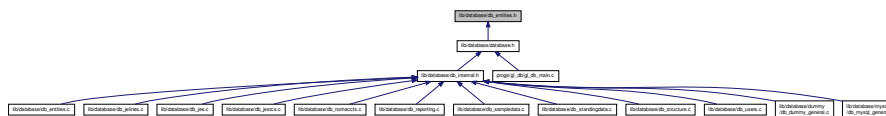
5.4 lib/database/db_entities.h File Reference

Interface to entities functionality.

```
#include <stdbool.h>
#include "datastruct/ds_str.h"
```



bioRxiv preprint doi: <https://doi.org/10.1101/2020.05.14.244400>; this version posted May 14, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.



- `bool db_create_entities_table` (void)
Creates the entities table in the database.
- `bool db_drop_entities_table` (void)
Drops the entities table in the database.
- `ds_str db_list_entities_report` (void)
Creates a report listing all entities.

Interface to entities functionality.

Paul Griffiths

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.4.2 Function Documentation

5.4.2.1 `bool db_create_entities_table (void)`

Creates the entities table in the database.

Returns

`true` on success, `false` on failure.

5.4.2.2 `bool db_drop_entities_table (void)`

Drops the entities table in the database.

Returns

`true` on success, `false` on failure.

5.4.2.3 `ds_str db_list_entities_report (void)`

Creates a report listing all entities.

Returns

A `ds_str` containing the report.

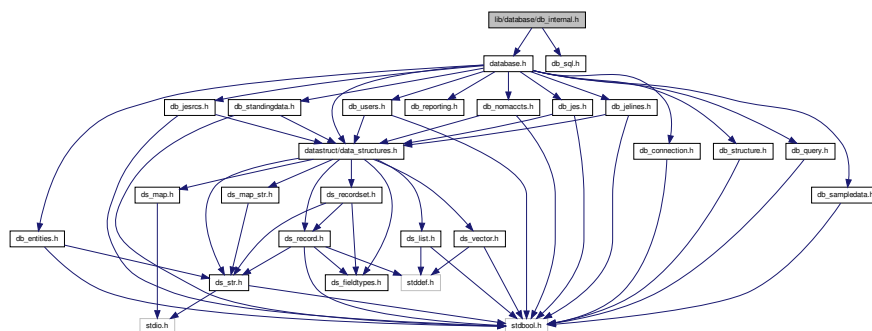
5.5 `lib/database/db_internal.h` File Reference

Internal library interface to database functionality.

```
#include "database.h"
```

```
#include "db_sql.h"
```

Include dependency graph for `db_internal.h`:



This graph shows which files directly or indirectly include this file:



5.5.1 Detailed Description

Internal library interface to database functionality. The library interface includes the individual SQL functions which should be encapsulated from the user.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

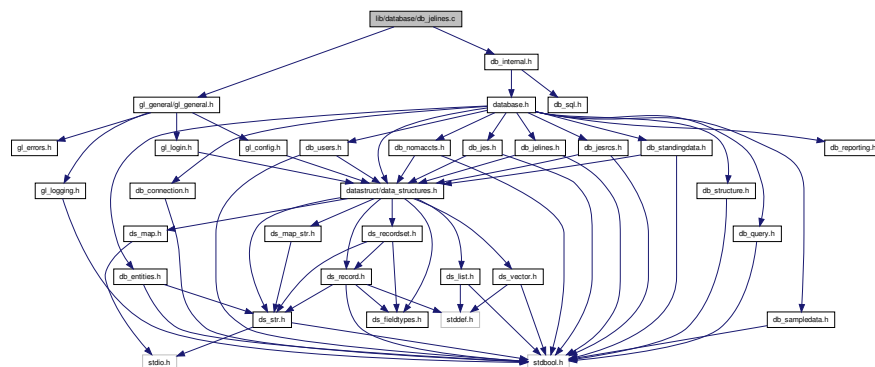
5.6 lib/database/db_jelines.c File Reference

Implementation of journal entries functionality.

```
#include "gl_general/gl_general.h"
```

```
#include "db_internal.h"
```

Include dependency graph for db_jelines.c:



Functions

- bool [db_create_jelines_table](#) (void)
Creates the journal entry lines table in the database.
- bool [db_drop_jelines_table](#) (void)
Drops the journal entry lines table from the database.
- [ds_str db_list_jelines_report](#) (void)
Creates a report listing all journal entry lines..

5.6.1 Detailed Description

Implementation of journal entries functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.6.2 Function Documentation

5.6.2.1 `bool db_create_jelines_table (void)`

Creates the journal entry lines table in the database.

Returns

`true` on success, `false` on failure.

5.6.2.2 `bool db_drop_jelines_table (void)`

Drops the journal entry lines table from the database.

Returns

`true` on success, `false` on failure.

5.6.2.3 `ds_str db_list_jelines_report (void)`

Creates a report listing all journal entry lines..

Returns

A `ds_str` containing the report.

5.7 `lib/database/db_jelines.h` File Reference

Interface to journal entry lines functionality.

```
#include <stdbool.h>
#include "datastruct/data_structures.h"
```

```

graph TD
    A["lib/database/db_jelines.h"] --> B["datastruct/data_structures.h"]
    B --> C["ds_list.h"]
    B --> D["ds_vector.h"]
    B --> E["ds_record.h"]
    B --> F["ds_recordset.h"]
    B --> G["ds_map_str.h"]
    B --> H["ds_map.h"]
    B --> I["ds_fieldtypes.h"]
    B --> J["ds_str.h"]
    B --> K["stdbool.h"]
    B --> L["stdio.h"]
    C --> K
    D --> K
    E --> K
    E --> I
    E --> J
    F --> E
    F --> I
    F --> J
    G --> J
    H --> L
    I --> J
    J --> L
    
```

[illegible]

- bool `db_create_jelines_table` (void)
Creates the journal entry lines table in the database.
- bool `db_drop_jelines_table` (void)
Drops the journal entry lines table from the database.
- `ds_str db_list_jelines_report` (void)
Creates a report listing all journal entry lines..

Paul Griffiths

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.7.2 Function Documentation

5.7.2.1 `bool db_create_jelines_table (void)`

Creates the journal entry lines table in the database.

Returns

`true` on success, `false` on failure.

5.7.2.2 `bool db_drop_jelines_table (void)`

Drops the journal entry lines table from the database.

Returns

`true` on success, `false` on failure.

5.7.2.3 `ds_str db_list_jelines_report (void)`

Creates a report listing all journal entry lines..

Returns

A [ds_str](#) containing the report.

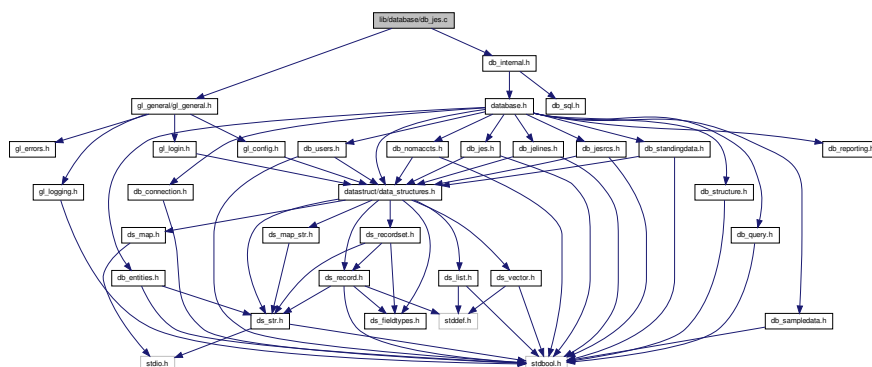
5.8 `lib/database/db_jes.c` File Reference

Implementation of journal entries functionality.

```
#include "gl_general/gl_general.h"
```

```
#include "db_internal.h"
```

Include dependency graph for `db_jes.c`:



Functions

- `bool db_create_jes_table (void)`
Creates the journal entries table in the database.
- `bool db_drop_jes_table (void)`

Drops the jes table from the database.

- [ds_str db_list_jes_report](#) (void)

Creates a report listing all journal entries.

5.8.1 Detailed Description

Implementation of journal entries functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.8.2 Function Documentation

5.8.2.1 `bool db_create_jes_table (void)`

Creates the journal entries table in the database.

Returns

`true` on success, `false` on failure.

5.8.2.2 `bool db_drop_jes_table (void)`

Drops the jes table from the database.

Returns

`true` on success, `false` on failure.

5.8.2.3 `ds_str db_list_jes_report (void)`

Creates a report listing all journal entries.

Returns

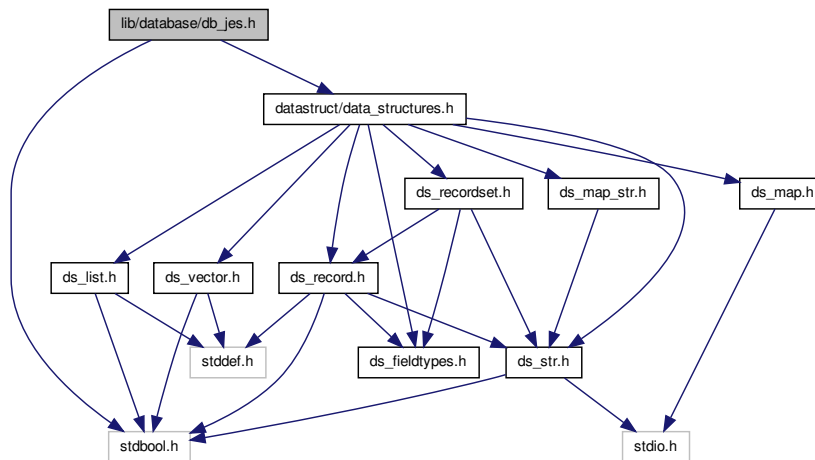
A [ds_str](#) containing the report.

5.9 lib/database/db_jes.h File Reference

Interface to journal entries functionality.

```
#include <stdbool.h>
#include "datastruct/data_structures.h"
```

Include dependency graph for `db_jes.h`:



This graph shows which files directly or indirectly include this file:



Functions

- `bool db_create_jes_table` (void)
Creates the journal entries table in the database.
- `bool db_drop_jes_table` (void)
Drops the jes table from the database.
- `ds_str db_list_jes_report` (void)
Creates a report listing all journal entries.

5.9.1 Detailed Description

Interface to journal entries functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

Drops the jesrcs table from the database.

- `ds_str db_list_jesrcs_report` (void)

Creates a report listing all journal entry sources.

5.10.1 Detailed Description

Implementation of journal entry sources functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.10.2 Function Documentation

5.10.2.1 `bool db_create_jesrcs_table (void)`

Creates the JE sources table in the database.

Returns

`true` on success, `false` on failure.

5.10.2.2 `bool db_drop_jesrcs_table (void)`

Drops the jesrcs table from the database.

Returns

`true` on success, `false` on failure.

5.10.2.3 `ds_str db_list_jesrcs_report (void)`

Creates a report listing all journal entry sources.

Returns

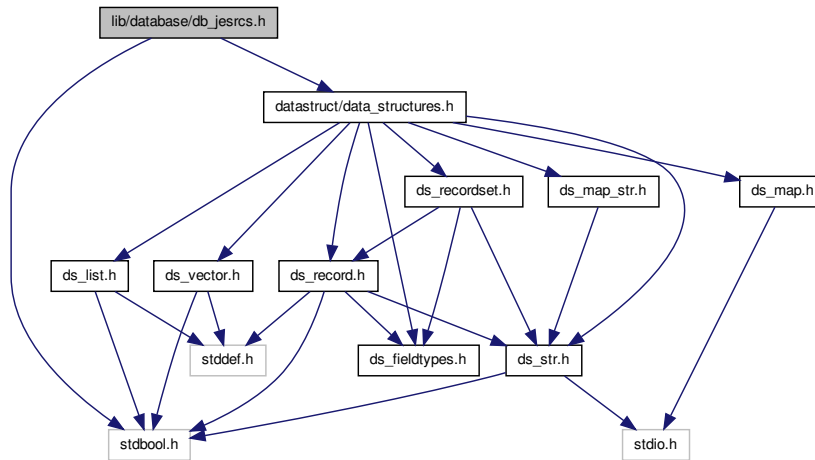
A `ds_str` containing the report.

5.11 lib/database/db_jesrcs.h File Reference

Interface to journal entry sources functionality.

```
#include <stdbool.h>
#include "datastruct/data_structures.h"
```

Include dependency graph for db_jesrcs.h:



This graph shows which files directly or indirectly include this file:



Functions

- bool [db_create_jesrcs_table](#) (void)
Creates the JE sources table in the database.
- bool [db_drop_jesrcs_table](#) (void)
Drops the jesrcs table from the database.
- [ds_str db_list_jesrcs_report](#) (void)
Creates a report listing all journal entry sources.

5.11.1 Detailed Description

Interface to journal entry sources functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.11.2 Function Documentation

5.11.2.1 `bool db_create_jesrcs_table (void)`

Creates the JE sources table in the database.

Returns

`true` on success, `false` on failure.

5.11.2.2 `bool db_drop_jesrcs_table (void)`

Drops the jesrcs table from the database.

Returns

`true` on success, `false` on failure.

5.11.2.3 `ds_str db_list_jesrcs_report (void)`

Creates a report listing all journal entry sources.

Returns

A [ds_str](#) containing the report.

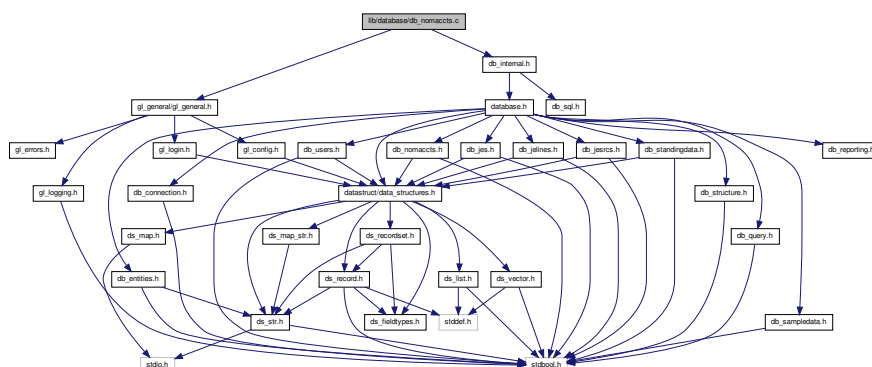
5.12 `lib/database/db_nomaccts.c` File Reference

Implementation of nominal accounts functionality.

```
#include "gl_general/gl_general.h"
```

```
#include "db_internal.h"
```

Include dependency graph for `db_nomaccts.c`:



Functions

- `bool db_create_nomaccts_table (void)`

Creates the nominal accounts table in the database.

- `bool db_drop_nomaccts_table (void)`

Drops the nomaccts table from the database.

- `ds_str db_list_nomaccts_report` (void)

Creates a report listing all nominal accounts.

5.12.1 Detailed Description

Implementation of nominal accounts functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.12.2 Function Documentation

5.12.2.1 `bool db_create_nomaccts_table (void)`

Creates the nominal accounts table in the database.

Returns

`true` on success, `false` on failure.

5.12.2.2 `bool db_drop_nomaccts_table (void)`

Drops the nomaccts table from the database.

Returns

`true` on success, `false` on failure.

5.12.2.3 `ds_str db_list_nomaccts_report (void)`

Creates a report listing all nominal accounts.

Returns

A `ds_str` containing the report.

5.13 lib/database/db_nomaccts.h File Reference

Interface to nominal accounts functionality.

```
#include <stdbool.h>
#include "datastruct/data_structures.h"
```


5.13.2 Function Documentation

5.13.2.1 `bool db_create_nomaccts_table (void)`

Creates the nominal accounts table in the database.

Returns

`true` on success, `false` on failure.

5.13.2.2 `bool db_drop_nomaccts_table (void)`

Drops the nomaccts table from the database.

Returns

`true` on success, `false` on failure.

5.13.2.3 `ds_str db_list_nomaccts_report (void)`

Creates a report listing all nominal accounts.

Returns

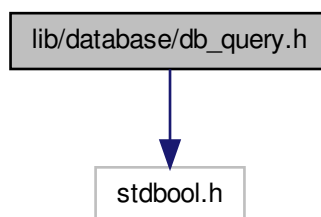
A `ds_str` containing the report.

5.14 lib/database/db_query.h File Reference

Interface to database query functionality.

```
#include <stdbool.h>
```

Include dependency graph for `db_query.h`:



This graph shows which files directly or indirectly include this file:



Functions

- bool `db_execute_query` (`ds_str` query)
Executes an SQL query on the database.

5.14.1 Detailed Description

Interface to database query functionality. Function implementations are provided by the individual database components.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.14.2 Function Documentation

5.14.2.1 bool `db_execute_query` (`ds_str` query)

Executes an SQL query on the database.

Parameters

<code>query</code>	The query to execute.
--------------------	-----------------------

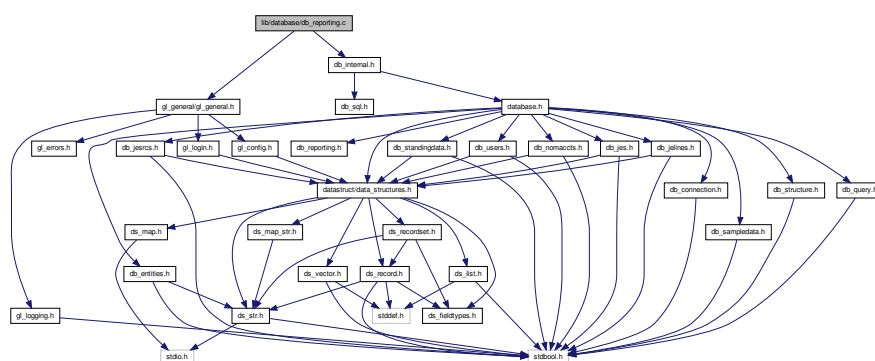
Returns

`true` if the query was successfully executed, `false` otherwise.

5.15 lib/database/db_reporting.c File Reference

Implementation of database reporting functionality.

```
#include "db_internal.h"
#include "gl_general/gl_general.h"
Include dependency graph for db_reporting.c:
```



Functions

- `ds_str db_create_report_from_query (ds_str query)`
Creates a text report from a query.
- `ds_recordset db_create_recordset_from_query (ds_str query)`
Creates a `ds_recordset` from a query.
- `ds_str db_current_trial_balance_report (ds_str entity)`
Runs the current trial balance report.

5.16.1 Detailed Description

Interface to database reporting functionality. Function implementations may be provided by the individual database components.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.16.2 Function Documentation

5.16.2.1 `ds_recordset db_create_recordset_from_query (ds_str query)`

Creates a `ds_recordset` from a query.

Parameters

<code>query</code>	The SELECT query to run.
--------------------	--------------------------

Returns

A `ds_recordset` containing the query result, or `NULL` on failure.

5.16.2.2 `ds_str db_create_report_from_query (ds_str query)`

Creates a text report from a query.

Parameters

<code>query</code>	The SELECT query to run.
--------------------	--------------------------

Returns

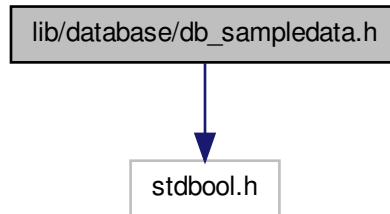
A `ds_str` containing the report, or `NULL` on failure.

5.16.2.3 `ds_str db_current_trial_balance_report (ds_str entity)`

Runs the current trial balance report.


```
#include <stdbool.h>
```

Include dependency graph for db_sampledata.h:



This graph shows which files directly or indirectly include this file:



Functions

- bool `db_load_sample_data` (void)
Loads sample data into the database.

5.18.1 Detailed Description

Interface to database sample data functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.19 lib/database/db_sql.h File Reference

Interface to database specific SQL strings.

This graph shows which files directly or indirectly include this file:



Functions

- const char * [db_create_users_table_sql](#) (void)
Returns the SQL query to create the users table.
- const char * [db_drop_users_table_sql](#) (void)
Returns the SQL query to drop the users table.
- const char * [db_list_users_report_sql](#) (void)
Returns the SQL query to run the "list users" report.
- const char * [db_create_entities_table_sql](#) (void)
Returns the SQL query to create the entities table.
- const char * [db_drop_entities_table_sql](#) (void)
Returns the SQL query to drop the entities table.
- const char * [db_list_entities_report_sql](#) (void)
Returns the SQL query to run the "list entities" report.
- const char * [db_create_jes_table_sql](#) (void)
Returns the SQL query to create the journal entries table.
- const char * [db_drop_jes_table_sql](#) (void)
Returns the SQL query to drop the journal entries table.
- const char * [db_list_jes_report_sql](#) (void)
Returns the SQL query to run the "list journal entries" report.
- const char * [db_create_nomaccts_table_sql](#) (void)
Returns the SQL query to create the nominal accounts table.
- const char * [db_drop_nomaccts_table_sql](#) (void)
Returns the SQL query to drop the nominal accounts table.
- const char * [db_list_nomaccts_report_sql](#) (void)
Returns the SQL query to run the "list nominal accounts" report.
- const char * [db_create_jelines_table_sql](#) (void)
Returns the SQL query to create the JE lines table.
- const char * [db_drop_jelines_table_sql](#) (void)
Returns the SQL query to drop the JE lines table.
- const char * [db_list_jelines_report_sql](#) (void)
Returns the SQL query to run the "list JE lines" report.
- const char * [db_current_trial_balance_report_sql](#) (void)
Returns the SQL query to run the "current TB" report.
- const char * [db_create_jesrcs_table_sql](#) (void)
Returns the SQL query to create the JE sources table.
- const char * [db_drop_jesrcs_table_sql](#) (void)
Returns the SQL query to drop the JE sources table.
- const char * [db_list_jesrcs_report_sql](#) (void)
Returns the SQL query to run the "list JE sources" report.
- const char * [db_create_standingdata_table_sql](#) (void)
Returns the SQL query to create the standing data table.
- const char * [db_drop_standingdata_table_sql](#) (void)
Returns the SQL query to drop the standing data table.
- const char * [db_show_standingdata_report_sql](#) (void)
Returns the SQL query to run the "show standing data" report.

5.19.1 Detailed Description

Interface to database specific SQL strings. Function implementations are provided by the individual database components.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.19.2 Function Documentation

5.19.2.1 `const char* db_create_entities_table_sql (void)`

Returns the SQL query to create the entities table.

Returns

The SQL query.

5.19.2.2 `const char* db_create_jelines_table_sql (void)`

Returns the SQL query to create the JE lines table.

Returns

The SQL query.

5.19.2.3 `const char* db_create_jes_table_sql (void)`

Returns the SQL query to create the journal entries table.

Returns

The SQL query.

5.19.2.4 `const char* db_create_jesrcs_table_sql (void)`

Returns the SQL query to create the JE sources table.

Returns

The SQL query.

5.19.2.5 `const char* db_create_nomaccts_table_sql (void)`

Returns the SQL query to create the nominal accounts table.

Returns

The SQL query.

5.19.2.6 `const char* db_create_standingdata_table_sql (void)`

Returns the SQL query to create the standing data table.

Returns

The SQL query.

5.19.2.7 `const char* db_create_users_table_sql (void)`

Returns the SQL query to create the users table.

Returns

The SQL query.

5.19.2.8 `const char* db_current_trial_balance_report_sql (void)`

Returns the SQL query to run the "current TB" report.

Returns

The SQL query.

5.19.2.9 `const char* db_drop_entities_table_sql (void)`

Returns the SQL query to drop the entities table.

Returns

The SQL query.

5.19.2.10 `const char* db_drop_jelines_table_sql (void)`

Returns the SQL query to drop the JE lines table.

Returns

The SQL query.

5.19.2.11 `const char* db_drop_jes_table_sql (void)`

Returns the SQL query to drop the journal entries table.

Returns

The SQL query.

5.19.2.12 `const char* db_drop_jesrcs_table_sql (void)`

Returns the SQL query to drop the JE sources table.

Returns

The SQL query.

5.19.2.13 const char* db_drop_nomaccts_table_sql (void)

Returns the SQL query to drop the nominal accounts table.

Returns

The SQL query.

5.19.2.14 const char* db_drop_standingdata_table_sql (void)

Returns the SQL query to drop the standing data table.

Returns

The SQL query.

5.19.2.15 const char* db_drop_users_table_sql (void)

Returns the SQL query to drop the users table.

Returns

The SQL query.

5.19.2.16 const char* db_list_entities_report_sql (void)

Returns the SQL query to run the "list entities" report.

Returns

The SQL query.

5.19.2.17 const char* db_list_jelines_report_sql (void)

Returns the SQL query to run the "list JE lines" report.

Returns

The SQL query.

5.19.2.18 const char* db_list_jes_report_sql (void)

Returns the SQL query to run the "list journal entries" report.

Returns

The SQL query.

5.19.2.19 const char* db_list_jesrcs_report_sql (void)

Returns the SQL query to run the "list JE sources" report.

Returns

The SQL query.

Returns the SQL query to run the "list nominal accounts" report.

The SQL query.

Returns the SQL query to run the "list users" report.

The SQL query.

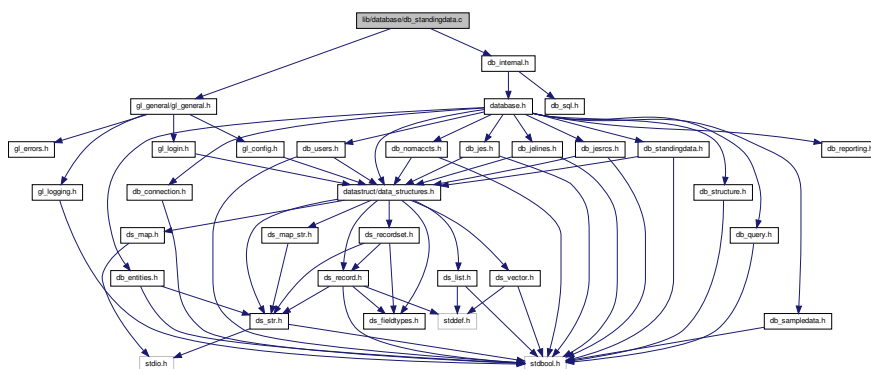
Returns the SQL query to run the "show standing data" report.

The SQL query.

Implementation of standing data functionality.

```
#include "db internal.h"
```

Include dependency graph for db standingdata.c:



- bool `db_create_standingdata_table` (void)
Creates the standing data table in the database.
- bool `db_drop_standingdata_table` (void)
Drops the standingdata table from the database.
- `ds_str db_show_standingdata_report` (void)
Creates a report showing standing data.

5.20.1 Detailed Description

Implementation of standing data functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.20.2 Function Documentation

5.20.2.1 `bool db_create_standingdata_table (void)`

Creates the standing data table in the database.

Returns

`true` on success, `false` on failure.

5.20.2.2 `bool db_drop_standingdata_table (void)`

Drops the standingdata table from the database.

Returns

`true` on success, `false` on failure.

5.20.2.3 `ds_str db_show_standingdata_report (void)`

Creates a report showing standing data.

Returns

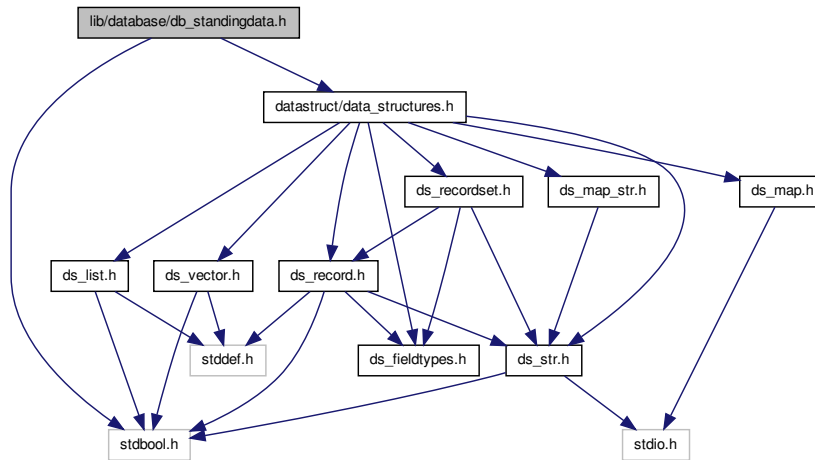
A `ds_str` containing the report.

5.21 `lib/database/db_standingdata.h` File Reference

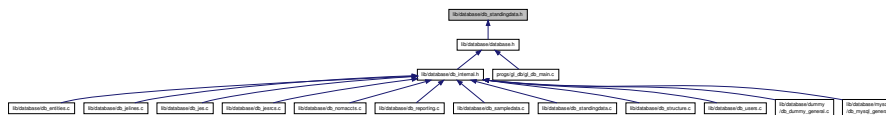
Interface to journal entries functionality.

```
#include <stdbool.h>
#include "datastruct/data_structures.h"
```

Include dependency graph for db_standingdata.h:



This graph shows which files directly or indirectly include this file:



Functions

- bool [db_create_standingdata_table](#) (void)
Creates the standing data table in the database.
- bool [db_drop_standingdata_table](#) (void)
Drops the standingdata table from the database.
- [ds_str db_show_standingdata_report](#) (void)
Creates a report showing standing data.

5.21.1 Detailed Description

Interface to journal entries functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.21.2 Function Documentation

5.21.2.1 `bool db_create_standingdata_table (void)`

Creates the standing data table in the database.

Returns

`true` on success, `false` on failure.

5.21.2.2 `bool db_drop_standingdata_table (void)`

Drops the standingdata table from the database.

Returns

`true` on success, `false` on failure.

5.21.2.3 `ds_str db_show_standingdata_report (void)`

Creates a report showing standing data.

Returns

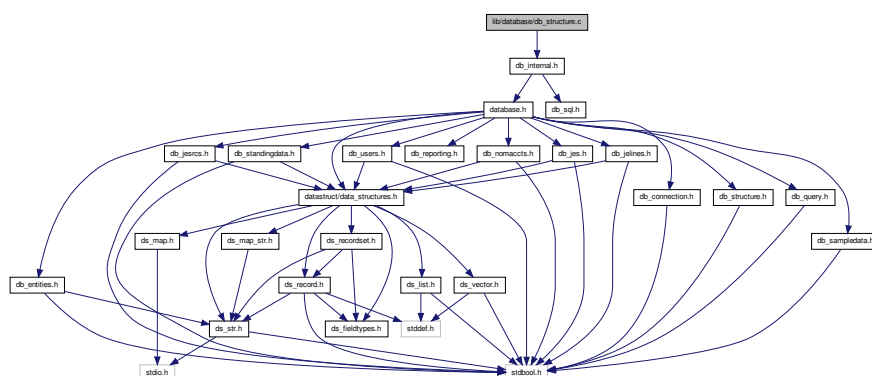
A `ds_str` containing the report.

5.22 `lib/database/db_structure.c` File Reference

Implementation of database structure functionality.

```
#include "db_internal.h"
```

Include dependency graph for `db_structure.c`:



Functions

- `bool db_create_database_structure (void)`
Creates an empty database structure.
- `bool db_delete_database_structure (void)`
Deletes the database structure.

5.22.1 Detailed Description

Implementation of database structure functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.22.2 Function Documentation

5.22.2.1 `bool db_create_database_structure (void)`

Creates an empty database structure.

Returns

`true` on success, `false` on failure.

5.22.2.2 `bool db_delete_database_structure (void)`

Deletes the database structure.

Returns

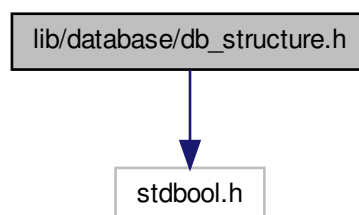
`true` on success, `false` on failure.

5.23 lib/database/db_structure.h File Reference

Interface to database structure functionality.

```
#include <stdbool.h>
```

Include dependency graph for db_structure.h:



This graph shows which files directly or indirectly include this file:



Functions

- bool [db_create_database_structure](#) (void)
Creates an empty database structure.
- bool [db_delete_database_structure](#) (void)
Deletes the database structure.

5.23.1 Detailed Description

Interface to database structure functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.23.2 Function Documentation

5.23.2.1 bool db_create_database_structure (void)

Creates an empty database structure.

Returns

true on success, false on failure.

5.23.2.2 bool db_delete_database_structure (void)

Deletes the database structure.

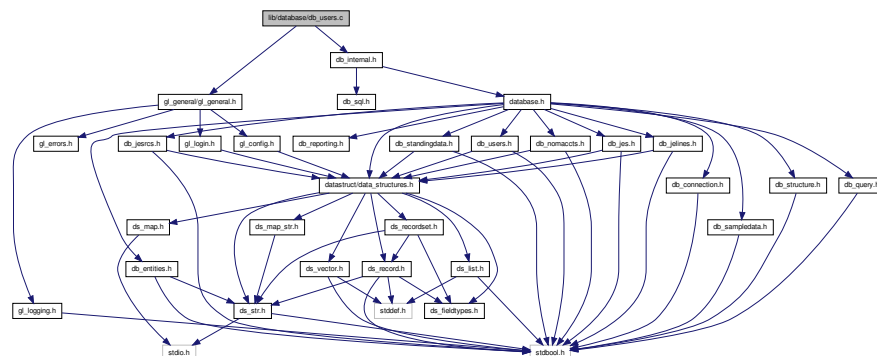
Returns

true on success, false on failure.

5.24 lib/database/db_users.c File Reference

Implementation of users functionality.


```
#include "db_internal.h"
#include "gl_general/gl_general.h"
Include dependency graph for db_users.c:
```



Functions

- bool `db_create_users_table` (void)
Creates the users table in the database.
- bool `db_drop_users_table` (void)
Drops the users table from the database.
- `ds_str db_list_users_report` (void)
Creates a report listing all users.

5.24.1 Detailed Description

Implementation of users functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.24.2 Function Documentation

5.24.2.1 bool db_create_users_table (void)

Creates the users table in the database.

Returns

true on success, false on failure.

5.24.2.2 `bool db_drop_users_table (void)`

Drops the users table from the database.

Returns

`true` on success, `false` on failure.

5.24.2.3 `ds_str db_list_users_report (void)`

Creates a report listing all users.

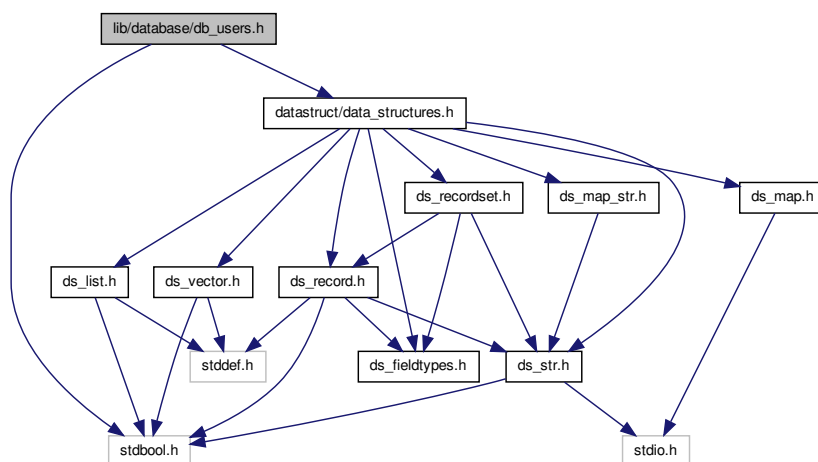
Returns

A `ds_str` containing the report.

5.25 `lib/database/db_users.h` File Reference

Interface to users functionality.

```
#include <stdbool.h>
#include "datastruct/data_structures.h"
Include dependency graph for db_users.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- `bool db_create_users_table (void)`

Creates the users table in the database.

- `bool db_drop_users_table` (void)

Drops the users table from the database.

- `ds_str db_list_users_report` (void)

Creates a report listing all users.

5.25.1 Detailed Description

Interface to users functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.25.2 Function Documentation

5.25.2.1 `bool db_create_users_table (void)`

Creates the users table in the database.

Returns

`true` on success, `false` on failure.

5.25.2.2 `bool db_drop_users_table (void)`

Drops the users table from the database.

Returns

`true` on success, `false` on failure.

5.25.2.3 `ds_str db_list_users_report (void)`

Creates a report listing all users.

Returns

A `ds_str` containing the report.

5.26 lib/database/dummy/db_dummy_create_entities_table_sql.c File Reference

Returns dummy SQL query to create entities table.

Functions

- `const char * db_create_entities_table_sql` (void)

Returns the SQL query to create the entities table.

5.26.1 Detailed Description

Returns dummy SQL query to create entities table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.26.2 Function Documentation

5.26.2.1 `const char* db_create_entities_table_sql (void)`

Returns the SQL query to create the entities table.

Returns

The SQL query.

5.27 lib/database/dummy/db_dummy_create_users_table_sql.c File Reference

Returns dummy SQL query to create users table.

Functions

- `const char * db_create_users_table_sql (void)`
Returns the SQL query to create the users table.

5.27.1 Detailed Description

Returns dummy SQL query to create users table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.27.2 Function Documentation

5.27.2.1 `const char* db_create_users_table_sql (void)`

Returns the SQL query to create the users table.

Returns

The SQL query.

5.28 lib/database/dummy/db_dummy_drop_entities_table_sql.c File Reference

Returns dummy SQL query to drop entities table.

Functions

- const char * [db_drop_entities_table_sql](#) (void)
Returns the SQL query to drop the entities table.

5.28.1 Detailed Description

Returns dummy SQL query to drop entities table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.28.2 Function Documentation

5.28.2.1 const char* db_drop_entities_table_sql (void)

Returns the SQL query to drop the entities table.

Returns

The SQL query.

5.29 lib/database/dummy/db_dummy_drop_users_table_sql.c File Reference

Returns dummy SQL query to drop users table.

Functions

- const char * [db_drop_users_table_sql](#) (void)
Returns the SQL query to drop the users table.

5.29.1 Detailed Description

Returns dummy SQL query to drop users table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.30.1 Detailed Description

Implementation of dummy database functionality. This module is useful when compiling for testing purpose on a system without any of the supported database development libraries available.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.30.2 Macro Definition Documentation

5.30.2.1 `#define _XOPEN_SOURCE 600`

UNIX feature test macro

5.30.3 Function Documentation

5.30.3.1 `bool db_connect (const char * host, const char * database, const char * username, const char * password)`

Connects to a database.

Parameters

<i>host</i>	The hostname.
<i>database</i>	The database name.
<i>username</i>	The username with which to connect.
<i>password</i>	The password for the specified user.

Returns

`true` if the connection was successfully made, `false` otherwise.

5.30.3.2 `ds_recordset db_create_recordset_from_query (ds_str query)`

Creates a [ds_recordset](#) from a query.

Parameters

<i>query</i>	The SELECT query to run.
--------------	--------------------------

Returns

A [ds_recordset](#) containing the query result, or `NULL` on failure.

5.30.3.3 `bool db_execute_query (ds_str query)`

Executes an SQL query on the database.

Parameters

<i>query</i>	The query to execute.
--------------	-----------------------

Returns

`true` if the query was successfully executed, `false` otherwise.

5.31 lib/database/dummy/db_dummy_list_entities_report_sql.c File Reference

Returns dummy SQL query to create list entities report.

Functions

- `const char * db_list_entities_report_sql (void)`
Returns the SQL query to run the "list entities" report.

5.31.1 Detailed Description

Returns dummy SQL query to create list entities report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.31.2 Function Documentation

5.31.2.1 `const char* db_list_entities_report_sql (void)`

Returns the SQL query to run the "list entities" report.

Returns

The SQL query.

5.32 lib/database/dummy/db_dummy_list_users_report_sql.c File Reference

Returns dummy SQL query to create list users report.

Functions

- `const char * db_list_users_report_sql (void)`
Returns the SQL query to run the "list users" report.

5.32.1 Detailed Description

Returns dummy SQL query to create list users report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.32.2 Function Documentation

5.32.2.1 `const char* db_list_users_report_sql (void)`

Returns the SQL query to run the "list users" report.

Returns

The SQL query.

5.33 lib/database/mysql/db_mysql_create_entities_table_sql.c File Reference

Returns MYSQL SQL query to create entities table.

Functions

- `const char * db_create_entities_table_sql (void)`
Returns the SQL query to create the entities table.

5.33.1 Detailed Description

Returns MYSQL SQL query to create entities table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.33.2 Function Documentation

5.33.2.1 `const char* db_create_entities_table_sql (void)`

Returns the SQL query to create the entities table.

Returns

The SQL query.

5.34 lib/database/mysql/db_mysql_create_jelines_table_sql.c File Reference

Returns MYSQL SQL query to create journal entry lines table.

Functions

- `const char * db_create_jelines_table_sql (void)`
Returns the SQL query to create the JE lines table.

5.34.1 Detailed Description

Returns MYSQL SQL query to create journal entry lines table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.34.2 Function Documentation

5.34.2.1 `const char* db_create_jelines_table_sql (void)`

Returns the SQL query to create the JE lines table.

Returns

The SQL query.

5.35 lib/database/mysql/db_mysql_create_jes_table_sql.c File Reference

Returns MYSQL SQL query to create journal entries table.

Functions

- `const char * db_create_jes_table_sql (void)`
Returns the SQL query to create the journal entries table.

5.35.1 Detailed Description

Returns MYSQL SQL query to create journal entries table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.35.2 Function Documentation

5.35.2.1 `const char* db_create_jes_table_sql (void)`

Returns the SQL query to create the journal entries table.

Returns

The SQL query.

5.36 lib/database/mysql/db_mysql_create_jesrcs_table_sql.c File Reference

Returns MYSQL SQL query to create JE sources table.

Functions

- `const char * db_create_jesrcs_table_sql (void)`
Returns the SQL query to create the JE sources table.

5.36.1 Detailed Description

Returns MYSQL SQL query to create JE sources table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.36.2 Function Documentation

5.36.2.1 `const char* db_create_jesrcs_table_sql (void)`

Returns the SQL query to create the JE sources table.

Returns

The SQL query.

5.37 lib/database/mysql/db_mysql_create_nomaccts_table_sql.c File Reference

Returns MYSQL SQL query to create nominal accounts table.

Functions

- `const char * db_create_nomaccts_table_sql (void)`
Returns the SQL query to create the nominal accounts table.

5.37.1 Detailed Description

Returns MYSQL SQL query to create nominal accounts table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.37.2 Function Documentation

5.37.2.1 `const char* db_create_nomaccts_table_sql (void)`

Returns the SQL query to create the nominal accounts table.

Returns

The SQL query.

5.38 lib/database/mysql/db_mysql_create_standingdata_table_sql.c File Reference

Returns MYSQL SQL query to create standing data table.

Functions

- `const char * db_create_standingdata_table_sql (void)`
Returns the SQL query to create the standing data table.

5.38.1 Detailed Description

Returns MYSQL SQL query to create standing data table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.38.2 Function Documentation

5.38.2.1 `const char* db_create_standingdata_table_sql (void)`

Returns the SQL query to create the standing data table.

Returns

The SQL query.

5.39 lib/database/mysql/db_mysql_create_users_table_sql.c File Reference

Returns MYSQL SQL query to create users table.

Functions

- const char * [db_create_users_table_sql](#) (void)
Returns the SQL query to create the users table.

5.39.1 Detailed Description

Returns MYSQL SQL query to create users table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.39.2 Function Documentation

5.39.2.1 const char* [db_create_users_table_sql](#) (void)

Returns the SQL query to create the users table.

Returns

The SQL query.

5.40 lib/database/mysql/db_mysql_current_trial_balance_report_sql.c File Reference

Returns MYSQL SQL query to create current TB report.

Functions

- const char * [db_current_trial_balance_report_sql](#) (void)
Returns the SQL query to run the "current TB" report.

5.40.1 Detailed Description

Returns MYSQL SQL query to create current TB report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.40.2 Function Documentation

5.40.2.1 `const char* db_current_trial_balance_report_sql (void)`

Returns the SQL query to run the "current TB" report.

Returns

The SQL query.

5.41 `lib/database/mysql/db_mysql_drop_entities_table_sql.c` File Reference

Returns MYSQL SQL query to drop entities table.

Functions

- `const char * db_drop_entities_table_sql (void)`
Returns the SQL query to drop the entities table.

5.41.1 Detailed Description

Returns MYSQL SQL query to drop entities table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.41.2 Function Documentation

5.41.2.1 `const char* db_drop_entities_table_sql (void)`

Returns the SQL query to drop the entities table.

Returns

The SQL query.

5.42 `lib/database/mysql/db_mysql_drop_jelines_table_sql.c` File Reference

Returns MYSQL SQL query to drop journal entry lines table.

Functions

- `const char * db_drop_jelines_table_sql (void)`
Returns the SQL query to drop the JE lines table.

5.42.1 Detailed Description

Returns MYSQL SQL query to drop journal entry lines table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.42.2 Function Documentation

5.42.2.1 `const char* db_drop_jelines_table_sql (void)`

Returns the SQL query to drop the JE lines table.

Returns

The SQL query.

5.43 lib/database/mysql/db_mysql_drop_jes_table_sql.c File Reference

Returns MYSQL SQL query to drop entities table.

Functions

- `const char * db_drop_jes_table_sql (void)`
Returns the SQL query to drop the journal entries table.

5.43.1 Detailed Description

Returns MYSQL SQL query to drop entities table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.43.2 Function Documentation

5.43.2.1 `const char* db_drop_jes_table_sql (void)`

Returns the SQL query to drop the journal entries table.

Returns

The SQL query.

5.44 lib/database/mysql/db_mysql_drop_jesrcs_table_sql.c File Reference

Returns MYSQL SQL query to drop JE sources table.

Functions

- const char * [db_drop_jesrcs_table_sql](#) (void)
Returns the SQL query to drop the JE sources table.

5.44.1 Detailed Description

Returns MYSQL SQL query to drop JE sources table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.44.2 Function Documentation

5.44.2.1 const char* db_drop_jesrcs_table_sql (void)

Returns the SQL query to drop the JE sources table.

Returns

The SQL query.

5.45 lib/database/mysql/db_mysql_drop_nomaccts_table_sql.c File Reference

Returns MYSQL SQL query to drop nominal accounts table.

Functions

- const char * [db_drop_nomaccts_table_sql](#) (void)
Returns the SQL query to drop the nominal accounts table.

5.45.1 Detailed Description

Returns MYSQL SQL query to drop nominal accounts table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.45.2 Function Documentation

5.45.2.1 `const char* db_drop_nomaccts_table_sql (void)`

Returns the SQL query to drop the nominal accounts table.

Returns

The SQL query.

5.46 lib/database/mysql/db_mysql_drop_standingdata_table_sql.c File Reference

Returns MYSQL SQL query to drop standing data table.

Functions

- `const char * db_drop_standingdata_table_sql (void)`
Returns the SQL query to drop the standing data table.

5.46.1 Detailed Description

Returns MYSQL SQL query to drop standing data table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.46.2 Function Documentation

5.46.2.1 `const char* db_drop_standingdata_table_sql (void)`

Returns the SQL query to drop the standing data table.

Returns

The SQL query.

5.47 lib/database/mysql/db_mysql_drop_users_table_sql.c File Reference

Returns MYSQL SQL query to drop users table.

Functions

- `const char * db_drop_users_table_sql (void)`
Returns the SQL query to drop the users table.

5.47.1 Detailed Description

Returns MYSQL SQL query to drop users table.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.47.2 Function Documentation

5.47.2.1 `const char* db_drop_users_table_sql (void)`

Returns the SQL query to drop the users table.

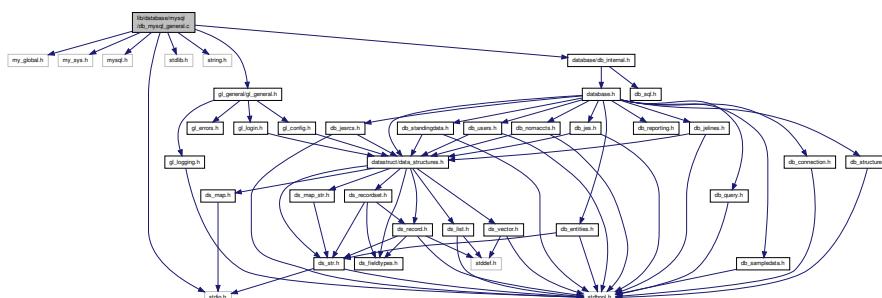
Returns

The SQL query.

5.48 lib/database/mysql/db_mysql_general.c File Reference

Implementation of MYSQL database functionality.

```
#include <my_global.h>
#include <my_sys.h>
#include <mysql.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "gl_general/gl_general.h"
#include "database/db_internal.h"
Include dependency graph for db_mysql_general.c:
```



Functions

- bool **db_connect** (const char *host, const char *database, const char *username, const char *password)
Connects to a database.
- void **db_close** (void)
Disconnects from a database.

- bool [db_execute_query](#) ([ds_str](#) query)
Executes an SQL query on the database.
- [ds_recordset](#) [db_create_recordset_from_query](#) ([ds_str](#) query)
Creates a [ds_recordset](#) from a query.

Variables

- MYSQL * [main_mss](#) = NULL
- MYSQL * [conn_mss](#) = NULL

5.48.1 Detailed Description

Implementation of MYSQL database functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.48.2 Function Documentation

5.48.2.1 bool [db_connect](#) (const char * *host*, const char * *database*, const char * *username*, const char * *password*)

Connects to a database.

Parameters

<i>host</i>	The hostname.
<i>database</i>	The database name.
<i>username</i>	The username with which to connect.
<i>password</i>	The password for the specified user.

Returns

`true` if the connection was successfully made, `false` otherwise.

5.48.2.2 [ds_recordset](#) [db_create_recordset_from_query](#) ([ds_str](#) *query*)

Creates a [ds_recordset](#) from a query.

Parameters

<i>query</i>	The SELECT query to run.
--------------	--------------------------

Returns

A [ds_recordset](#) containing the query result, or `NULL` on failure.

5.48.2.3 `bool db_execute_query (ds_str query)`

Executes an SQL query on the database.

Parameters

<code>query</code>	The query to execute.
--------------------	-----------------------

Returns

`true` if the query was successfully executed, `false` otherwise.

5.48.3 Variable Documentation

5.48.3.1 `MYSQL* conn_mss = NULL`

MYSQL connection object.

5.48.3.2 `MYSQL* main_mss = NULL`

MYSQL initialization object.

5.49 `lib/database/mysql/db_mysql_list_entities_report_sql.c` File Reference

Returns MYSQL SQL query to create list entities report.

Functions

- `const char * db_list_entities_report_sql (void)`
Returns the SQL query to run the "list entities" report.

5.49.1 Detailed Description

Returns MYSQL SQL query to create list entities report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.49.2 Function Documentation

5.49.2.1 `const char* db_list_entities_report_sql (void)`

Returns the SQL query to run the "list entities" report.

Returns

The SQL query.

5.50 lib/database/mysql/db_mysql_list_jelines_report_sql.c File Reference

Returns MYSQL SQL query to create JE lines report.

Functions

- const char * [db_list_jelines_report_sql](#) (void)
Returns the SQL query to run the "list JE lines" report.

5.50.1 Detailed Description

Returns MYSQL SQL query to create JE lines report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.50.2 Function Documentation

5.50.2.1 const char* db_list_jelines_report_sql (void)

Returns the SQL query to run the "list JE lines" report.

Returns

The SQL query.

5.51 lib/database/mysql/db_mysql_list_jes_report_sql.c File Reference

Returns MYSQL SQL query to create journal entries report.

Functions

- const char * [db_list_jes_report_sql](#) (void)
Returns the SQL query to run the "list journal entries" report.

5.51.1 Detailed Description

Returns MYSQL SQL query to create journal entries report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.51.2 Function Documentation

5.51.2.1 `const char* db_list_jes_report_sql (void)`

Returns the SQL query to run the "list journal entries" report.

Returns

The SQL query.

5.52 `lib/database/mysql/db_mysql_list_jesrcs_report_sql.c` File Reference

Returns MYSQL SQL query to create JE sources report.

Functions

- `const char * db_list_jesrcs_report_sql (void)`
Returns the SQL query to run the "list JE sources" report.

5.52.1 Detailed Description

Returns MYSQL SQL query to create JE sources report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.52.2 Function Documentation

5.52.2.1 `const char* db_list_jesrcs_report_sql (void)`

Returns the SQL query to run the "list JE sources" report.

Returns

The SQL query.

5.53 `lib/database/mysql/db_mysql_list_nomaccts_report_sql.c` File Reference

Returns MYSQL SQL query to create list nominal accounts report.

Functions

- `const char * db_list_nomaccts_report_sql (void)`
Returns the SQL query to run the "list nominal accounts" report.

5.53.1 Detailed Description

Returns MYSQL SQL query to create list nominal accounts report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.53.2 Function Documentation

5.53.2.1 `const char* db_list_nomaccts_report_sql (void)`

Returns the SQL query to run the "list nominal accounts" report.

Returns

The SQL query.

5.54 lib/database/mysql/db_mysql_list_users_report_sql.c File Reference

Returns MYSQL SQL query to create list users report.

Functions

- `const char * db_list_users_report_sql (void)`
Returns the SQL query to run the "list users" report.

5.54.1 Detailed Description

Returns MYSQL SQL query to create list users report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.54.2 Function Documentation

5.54.2.1 `const char* db_list_users_report_sql (void)`

Returns the SQL query to run the "list users" report.

Returns

The SQL query.

5.55 lib/database/mysql/db_mysql_show_standingdata_report_sql.c File Reference

Returns MYSQL SQL query to create show standing data report.

Functions

- const char * [db_show_standingdata_report_sql](#) (void)

Returns the SQL query to run the "show standing data" report.

5.55.1 Detailed Description

Returns MYSQL SQL query to create show standing data report.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.55.2 Function Documentation

5.55.2.1 const char* db_show_standingdata_report_sql (void)

Returns the SQL query to run the "show standing data" report.

Returns

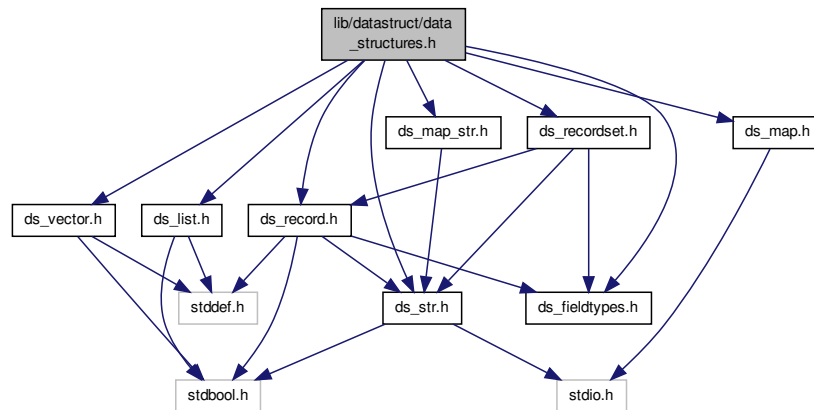
The SQL query.

5.56 lib/datastruct/data_structures.h File Reference

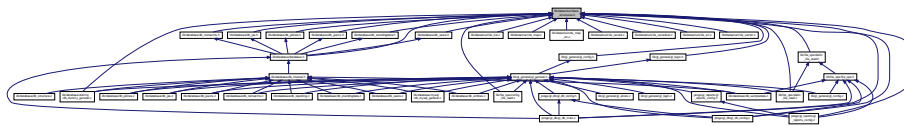
Interface to data structures.

```
#include "ds_list.h"
#include "ds_vector.h"
#include "ds_str.h"
#include "ds_map.h"
#include "ds_map_str.h"
#include "ds_fieldtypes.h"
#include "ds_record.h"
#include "ds_recordset.h"
```


Include dependency graph for data_structures.h:



This graph shows which files directly or indirectly include this file:



5.56.1 Detailed Description

Interface to data structures.

Author

Paul Griffiths

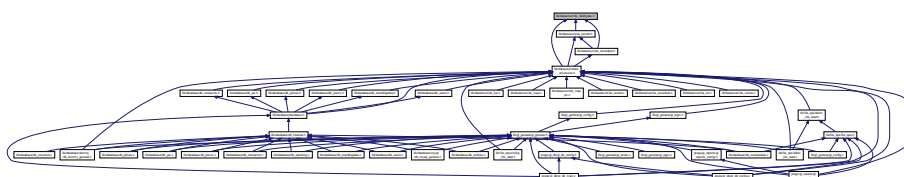
Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.57 lib/datastruct/ds_fieldtypes.h File Reference

Record field types enumeration.

This graph shows which files directly or indirectly include this file:



Enumerations

- enum `ds_field_types` { `DS_FIELD_STRING`, `DS_FIELD_INT`, `DS_FIELD_BOOLEAN`, `DS_FIELD_DOUBLE` }

5.57.1 Detailed Description

Record field types enumeration.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.57.2 Enumeration Type Documentation

5.57.2.1 enum `ds_field_types`

Enumeration for field type

Enumerator:

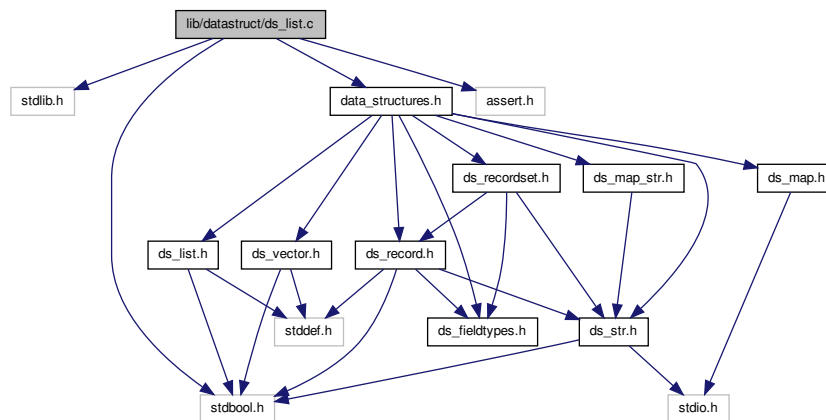
- `DS_FIELD_STRING`** Field is string type
- `DS_FIELD_INT`** Field is integer type
- `DS_FIELD_BOOLEAN`** Field is boolean type
- `DS_FIELD_DOUBLE`** Field is double type

5.58 lib/datastruct/ds_list.c File Reference

Implementation of generic doubly-linked list data structure.

```
#include <stdlib.h>
#include <stdbool.h>
#include <assert.h>
#include "data_structures.h"
```

Include dependency graph for `ds_list.c`:



Data Structures

- struct [ds_list_element](#)
- struct [ds_list](#)

Functions

- [ds_list ds_list_create](#) (const bool free_on_delete, void(*destructor)(void *))
Creates a new list.
- void [ds_list_destroy](#) ([ds_list](#) list)
Destroys a list and frees any associated resources.
- void [ds_list_destructor](#) (void *list)
A list destructor function.
- [ds_list ds_list_append](#) ([ds_list](#) list, void *data)
Appends an element to a list.
- void [ds_list_remove_tail](#) ([ds_list](#) list)
Removes the last element of a list.
- void [ds_list_remove_all](#) ([ds_list](#) list)
Removes all the elements from a list.
- void * [ds_list_element](#) ([ds_list](#) list, const size_t index)
Retrieves the data at a specified index.
- size_t [ds_list_length](#) ([ds_list](#) list)
Returns the number of elements in a list.
- bool [ds_list_is_empty](#) ([ds_list](#) list)
Checks if a list is empty.
- void [ds_list_seek_start](#) ([ds_list](#) list)
Sets the current element to the first element of a list.
- void [ds_list_seek_end](#) ([ds_list](#) list)
Sets the current element to the last element of a list.
- void * [ds_list_get_next_data](#) ([ds_list](#) list)
Returns the next element of the list.
- void * [ds_list_get_prev_data](#) ([ds_list](#) list)
Returns the previous element of the list.

5.58.1 Detailed Description

Implementation of generic doubly-linked list data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.58.2 Function Documentation

5.58.2.1 `ds_list ds_list_append (ds_list list, void * element)`

Appends an element to a list.

Parameters

<i>list</i>	The list to which to append.
<i>element</i>	The element to append.

Returns

The same list, or `NULL` on failure.

5.58.2.2 `ds_list ds_list_create (const bool free_on_delete, void(*)(void *) destructor)`

Creates a new list.

Parameters

<i>free_on_delete</i>	Set to <code>true</code> if the list elements should be destroyed when removed from the list, and when the list itself is destroyed. If set to <code>false</code> , the caller is responsible for destroying the elements prior to destroying the list.
<i>destructor</i>	Pointer to a destructor function to use for destroying the list elements, when <code>free_on_delete</code> is true. If this is set to <code>NULL</code> , <code>free()</code> from the standard C library will be used to destroy the elements.

Returns

A newly created list, or `NULL` on failure.

5.58.2.3 `void ds_list_destroy (ds_list list)`

Destroys a list and frees any associated resources.

Parameters

<i>list</i>	The list to destroy.
-------------	----------------------

5.58.2.4 `void ds_list_destructor (void * list)`

A list destructor function.

This function may be passed to `ds_list_create()` when creating a list of lists. It calls `ds_list_destroy()`, but the parameter of `ds_list_destroy()` is not compatible with the function signature expected by `ds_list_create()`, so this function provides an appropriate interface.

Parameters

<i>list</i>	The list to destroy.
-------------	----------------------

5.58.2.5 void* ds_list_element (ds_list list, const size_t index)

Retrieves the data at a specified index.

Parameters

<i>list</i>	The list from which to retrieve.
<i>index</i>	The index of the desired element.

Returns

A pointer to the data, or `NULL` if the index is out of range.

5.58.2.6 void* ds_list_get_next_data (ds_list list)

Returns the next element of the list.

This function returns the data of the "current element", and advances the current element pointer. Subsequent calls to this function will return successive elements.

Parameters

<i>list</i>	The list.
-------------	-----------

Returns

A pointer to the next element, or `NULL` if the end of the list has been reached.

5.58.2.7 void* ds_list_get_prev_data (ds_list list)

Returns the previous element of the list.

This function returns the data of the "current element", and decrements the current element pointer. Subsequent calls to this function will return successively earlier elements.

Parameters

<i>list</i>	The list.
-------------	-----------

Returns

A pointer to the previous element, or `NULL` if the start of the list has been reached.

5.58.2.8 bool ds_list_is_empty (ds_list list)

Checks if a list is empty.

Parameters

<i>list</i>	The list to check.
-------------	--------------------

Returns

`true` if the list is empty, `false` otherwise.

5.58.2.9 `size_t ds_list_length (ds_list list)`

Returns the number of elements in a list.

Parameters

<i>list</i>	The list.
-------------	-----------

Returns

The number of elements in the list.

5.58.2.10 `void ds_list_remove_all (ds_list list)`

Removes all the elements from a list.

Parameters

<i>list</i>	The list from which to remove.
-------------	--------------------------------

5.58.2.11 `void ds_list_remove_tail (ds_list list)`

Removes the last element of a list.

Parameters

<i>list</i>	The list from which to remove.
-------------	--------------------------------

5.58.2.12 `void ds_list_seek_end (ds_list list)`

Sets the current element to the last element of a list.

Parameters

<i>list</i>	The list.
-------------	-----------

5.58.2.13 `void ds_list_seek_start (ds_list list)`

Sets the current element to the first element of a list.

Parameters

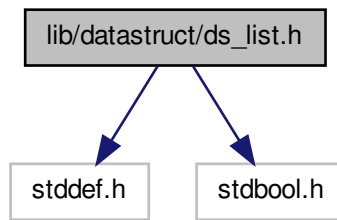
<i>list</i>	The list.
-------------	-----------

5.59 `lib/datastruct/ds_list.h` File Reference

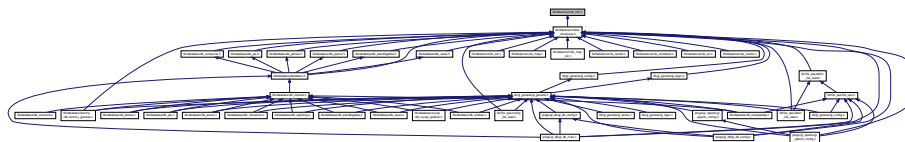
Interface to generic doubly-linked list data structure.

```
#include <stddef.h>
#include <stdbool.h>
```

Include dependency graph for ds_list.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct [ds_list](#) * [ds_list](#)

Functions

- [ds_list ds_list_create](#) (const bool free_on_delete, void(*destructor)(void *))
Creates a new list.
- void [ds_list_destroy](#) ([ds_list](#) list)
Destroys a list and frees any associated resources.
- void [ds_list_destructor](#) (void *list)
A list destructor function.
- [ds_list ds_list_append](#) ([ds_list](#) list, void *element)
Appends an element to a list.
- void [ds_list_remove_tail](#) ([ds_list](#) list)
Removes the last element of a list.
- void [ds_list_remove_all](#) ([ds_list](#) list)
Removes all the elements from a list.
- void * [ds_list_element](#) ([ds_list](#) list, const size_t index)
Retrieves the data at a specified index.
- size_t [ds_list_length](#) ([ds_list](#) list)
Returns the number of elements in a list.
- bool [ds_list_is_empty](#) ([ds_list](#) list)
Checks if a list is empty.
- void [ds_list_seek_start](#) ([ds_list](#) list)
Sets the current element to the first element of a list.
- void [ds_list_seek_end](#) ([ds_list](#) list)

Sets the current element to the last element of a list.

- void * [ds_list_get_next_data](#) (ds_list list)

Returns the next element of the list.

- void * [ds_list_get_prev_data](#) (ds_list list)

Returns the previous element of the list.

5.59.1 Detailed Description

Interface to generic doubly-linked list data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.59.2 Typedef Documentation

5.59.2.1 typedef struct ds_list* ds_list

Typedef for opaque list datatype

5.59.3 Function Documentation

5.59.3.1 ds_list ds_list_append (ds_list list, void * element)

Appends an element to a list.

Parameters

<i>list</i>	The list to which to append.
<i>element</i>	The element to append.

Returns

The same list, or `NULL` on failure.

5.59.3.2 ds_list ds_list_create (const bool free_on_delete, void(*) (void *) destructor)

Creates a new list.

Parameters

<i>free_on_delete</i>	Set to <code>true</code> if the list elements should be destroyed when removed from the list, and when the list itself is destroyed. If set to <code>false</code> , the caller is responsible for destroying the elements prior to destroying the list.
<i>destructor</i>	Pointer to a destructor function to use for destroying the list elements, when <code>free_on_delete</code> is true. If this is set to <code>NULL</code> , <code>free()</code> from the standard C library will be used to destroy the elements.

Returns

A newly created list, or `NULL` on failure.

5.59.3.3 void ds_list_destroy (ds_list list)

Destroys a list and frees any associated resources.

Parameters

<i>list</i>	The list to destroy.
-------------	----------------------

5.59.3.4 void ds_list_destructor (void * list)

A list destructor function.

This function may be passed to `ds_list_create()` when creating a list of lists. It calls `ds_list_destroy()`, but the parameter of `ds_list_destroy()` is not compatible with the function signature expected by `ds_list_create()`, so this function provides an appropriate interface.

Parameters

<i>list</i>	The list to destroy.
-------------	----------------------

5.59.3.5 void* ds_list_element (ds_list list, const size_t index)

Retrieves the data at a specified index.

Parameters

<i>list</i>	The list from which to retrieve.
<i>index</i>	The index of the desired element.

Returns

A pointer to the data, or `NULL` if the index is out of range.

5.59.3.6 void* ds_list_get_next_data (ds_list list)

Returns the next element of the list.

This function returns the data of the "current element", and advances the current element pointer. Subsequent calls to this function will return successive elements.

Parameters

<i>list</i>	The list.
-------------	-----------

Returns

A pointer to the next element, or `NULL` if the end of the list has been reached.

5.59.3.7 void* ds_list_get_prev.data (ds_list list)

Returns the previous element of the list.

This function returns the data of the "current element", and decrements the current element pointer. Subsequent calls to this function will return successively earlier elements.

Parameters

<i>list</i>	The list.
-------------	-----------

Returns

A pointer to the previous element, or `NULL` if the start of the list has been reached.

5.59.3.8 bool ds_list_is_empty (ds_list list)

Checks if a list is empty.

Parameters

<i>list</i>	The list to check.
-------------	--------------------

Returns

`true` if the list is empty, `false` otherwise.

5.59.3.9 size_t ds_list_length (ds_list list)

Returns the number of elements in a list.

Parameters

<i>list</i>	The list.
-------------	-----------

Returns

The number of elements in the list.

5.59.3.10 void ds_list_remove_all (ds_list list)

Removes all the elements from a list.

Parameters

<i>list</i>	The list from which to remove.
-------------	--------------------------------

5.59.3.11 void ds_list_remove_tail (ds_list list)

Removes the last element of a list.

Parameters

<i>list</i>	The list from which to remove.
-------------	--------------------------------

5.59.3.12 void ds_list_seek_end (ds_list list)

Sets the current element to the last element of a list.

Parameters

<i>list</i>	The list.
-------------	-----------

5.59.3.13 void ds_list_seek_start (ds_list list)

Sets the current element to the first element of a list.

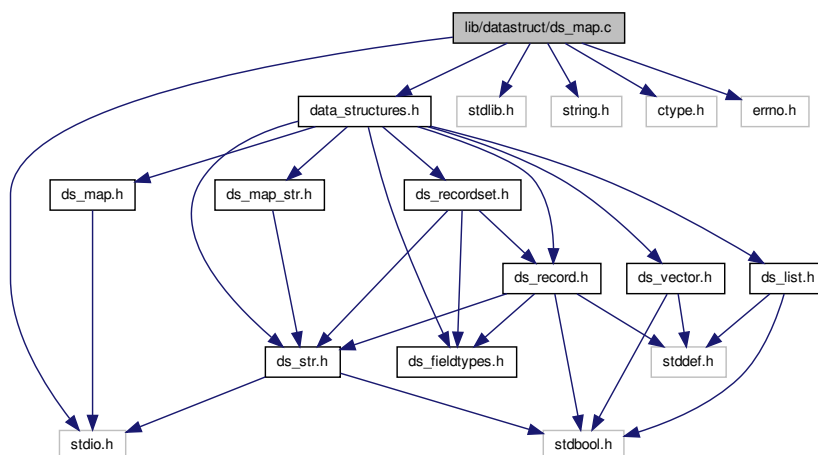
Parameters

<i>list</i>	The list.
-------------	-----------

5.60 lib/datastruct/ds_map.c File Reference

Implementation of string-string hash map data structure.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <errno.h>
#include "data_structures.h"
Include dependency graph for ds_map.c:
```



Data Structures

- struct [kv_pair_node](#)
- struct [ds_map](#)

Macros

- `#define _POSIX_C_SOURCE 200809L`
Enables POSIX library functions.

Functions

- [ds_map ds_map_init](#) (const size_t hash_size)
Initializes a hash map.
- void [ds_map_destroy](#) ([ds_map](#) map)
Destroys a hash map.
- const char * [ds_map_get_value](#) ([ds_map](#) map, const char *key)
Retrieves a value associated with a key in the map.
- void [ds_map_insert](#) ([ds_map](#) map, const char *key, const char *value)
Inserts a key-value pair into a map.
- void [ds_map_print_all](#) ([ds_map](#) map, FILE *outfile)
Prints all the key-value pairs in a map to stdout.

5.60.1 Detailed Description

Implementation of string-string hash map data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.60.2 Function Documentation

5.60.2.1 void ds_map_destroy (ds_map map)

Destroys a hash map.

Parameters

<i>map</i>	A reference to the map to destroy.
------------	------------------------------------

5.60.2.2 const char* ds_map_get_value (ds_map map, const char * key)

Retrieves a value associated with a key in the map.

Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.

Returns

A pointer to the value associated with the key, or `NULL` if the key is not in the map. The caller should not modify the string to which this pointer points.

5.60.2.3 `ds_map ds_map_init (const size_t hash_size)`

Initializes a hash map.

Parameters

<i>hash_size</i>	The number of possible hash values.
------------------	-------------------------------------

Returns

A reference to the newly-created hash map.

5.60.2.4 `void ds_map_insert (ds_map map, const char * key, const char * value)`

Inserts a key-value pair into a map.

The key and value are copied, so the caller may modify or `free()` them after calling this function.

Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.
<i>value</i>	The value.

5.60.2.5 `void ds_map_print_all (ds_map map, FILE * outfile)`

Prints all the key-value pairs in a map to stdout.

Parameters

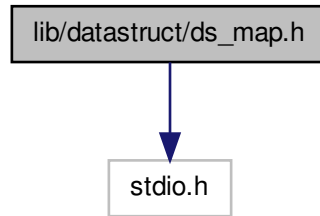
<i>map</i>	A reference to the map.
<i>outfile</i>	A FILE pointer to which to print the output.

5.61 lib/datastruct/ds_map.h File Reference

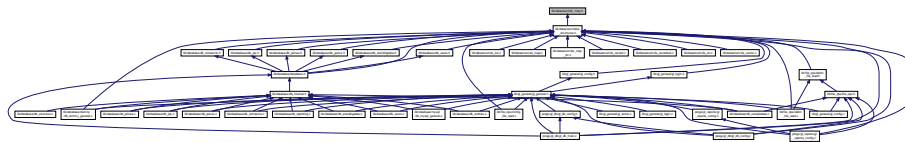
Interface to string-string hash map data structure.

```
#include <stdio.h>
```

Include dependency graph for ds_map.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct [ds_map](#) * [ds_map](#)

Functions

- [ds_map](#) [ds_map_init](#) (const size_t hash_size)
Initializes a hash map.
- void [ds_map_destroy](#) ([ds_map](#) map)
Destroys a hash map.
- const char * [ds_map_get_value](#) ([ds_map](#) map, const char *key)
Retrieves a value associated with a key in the map.
- void [ds_map_insert](#) ([ds_map](#) map, const char *key, const char *value)
Inserts a key-value pair into a map.
- void [ds_map_print_all](#) ([ds_map](#) map, FILE *outfile)
Prints all the key-value pairs in a map to stdout.

5.61.1 Detailed Description

Interface to string-string hash map data structure.

Author

Paul Griffiths

Copyright

Copyright 2013 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.61.2 Typedef Documentation

5.61.2.1 typedef struct ds_map* ds_map

Opaque data type for hash map

5.61.3 Function Documentation

5.61.3.1 void ds_map_destroy (ds_map map)

Destroys a hash map.

Parameters

<i>map</i>	A reference to the map to destroy.
------------	------------------------------------

5.61.3.2 const char* ds_map_get_value (ds_map map, const char * key)

Retrieves a value associated with a key in the map.

Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.

Returns

A pointer to the value associated with the key, or `NULL` if the key is not in the map. The caller should not modify the string to which this pointer points.

5.61.3.3 ds_map ds_map_init (const size_t hash_size)

Initializes a hash map.

Parameters

<i>hash_size</i>	The number of possible hash values.
------------------	-------------------------------------

Returns

A reference to the newly-created hash map.

5.61.3.4 void ds_map_insert (ds_map map, const char * key, const char * value)

Inserts a key-value pair into a map.

The key and value are copied, so the caller may modify or `free()` them after calling this function.

Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.
<i>value</i>	The value.

5.61.3.5 void ds_map_print_all (ds_map map, FILE * outfile)

Prints all the key-value pairs in a map to stdout.

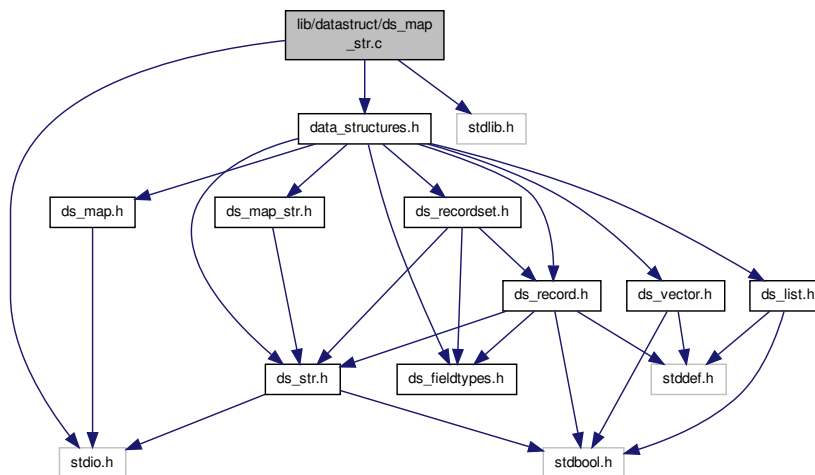
Parameters

<i>map</i>	A reference to the map.
<i>outfile</i>	A FILE pointer to which to print the output.

5.62 lib/datastruct/ds_map_str.c File Reference

Implementation of string-string hash map data structure.

```
#include <stdio.h>
#include <stdlib.h>
#include "data_structures.h"
Include dependency graph for ds_map_str.c:
```



Data Structures

- struct [kv_pair_node](#)
- struct [ds_map_str](#)

Functions

- [ds_map_str ds_map_str_init](#) (const size_t hash_size)
Initializes a hash map.
- void [ds_map_str_destroy](#) (ds_map_str map)
Destroys a hash map.
- [ds_str ds_map_str_get_value](#) (ds_map_str map, ds_str key)
Retrieves a value associated with a key in the map.
- void [ds_map_str_insert](#) (ds_map_str map, ds_str key, ds_str value)
Inserts a key-value pair into a map.

5.62.1 Detailed Description

Implementation of string-string hash map data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.62.2 Function Documentation

5.62.2.1 void ds_map_str_destroy (ds_map_str map)

Destroys a hash map.

Parameters

<i>map</i>	A reference to the map to destroy.
------------	------------------------------------

5.62.2.2 ds_str ds_map_str_get_value (ds_map_str map, ds_str key)

Retrieves a value associated with a key in the map.

Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.

Returns

A pointer to the value associated with the key, or `NULL` if the key is not in the map. The caller should not modify the string to which this pointer points.

5.62.2.3 ds_map_str ds_map_str_init (const size_t hash_size)

Initializes a hash map.

Parameters

<i>hash_size</i>	The number of possible hash values.
------------------	-------------------------------------

Returns

A reference to the newly-created hash map.

5.62.2.4 void ds_map_str_insert (ds_map_str map, ds_str key, ds_str value)

Inserts a key-value pair into a map.

The key and value are copied, so the caller may modify or `free()` them after calling this function.

Parameters

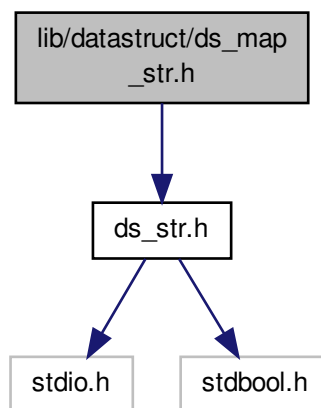
<i>map</i>	A reference to the hash map.
<i>key</i>	The key.
<i>value</i>	The value.

5.63 lib/datastruct/ds_map_str.h File Reference

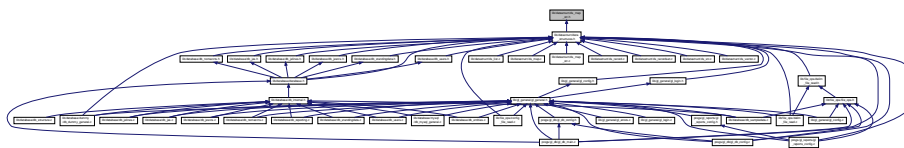
Interface to string-string hash map data structure.

```
#include "ds_str.h"
```

Include dependency graph for ds_map_str.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct [ds_map_str](#) * [ds_map_str](#)

Functions

- [ds_map_str ds_map_str_init](#) (const [size_t](#) hash_size)
Initializes a hash map.
- void [ds_map_str_destroy](#) ([ds_map_str](#) map)
Destroys a hash map.
- [ds_str ds_map_str_get_value](#) ([ds_map_str](#) map, [ds_str](#) key)
Retrieves a value associated with a key in the map.

- void `ds_map_str_insert` (`ds_map_str` map, `ds_str` key, `ds_str` value)
Inserts a key-value pair into a map.

5.63.1 Detailed Description

Interface to string-string hash map data structure.

Author

Paul Griffiths

Copyright

Copyright 2013 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.63.2 Typedef Documentation

5.63.2.1 typedef struct `ds_map_str*` `ds_map_str`

Opaque data type for hash map

5.63.3 Function Documentation

5.63.3.1 void `ds_map_str_destroy` (`ds_map_str` map)

Destroys a hash map.

Parameters

<i>map</i>	A reference to the map to destroy.
------------	------------------------------------

5.63.3.2 `ds_str` `ds_map_str_get_value` (`ds_map_str` map, `ds_str` key)

Retrieves a value associated with a key in the map.

Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.

Returns

A pointer to the value associated with the key, or `NULL` if the key is not in the map. The caller should not modify the string to which this pointer points.

5.63.3.3 `ds_map_str` `ds_map_str_init` (`const size_t` hash_size)

Initializes a hash map.

Parameters

<i>hash_size</i>	The number of possible hash values.
------------------	-------------------------------------

Returns

A reference to the newly-created hash map.

5.63.3.4 void ds_map_str_insert (ds_map_str map, ds_str key, ds_str value)

Inserts a key-value pair into a map.

The key and value are copied, so the caller may modify or `free()` them after calling this function.

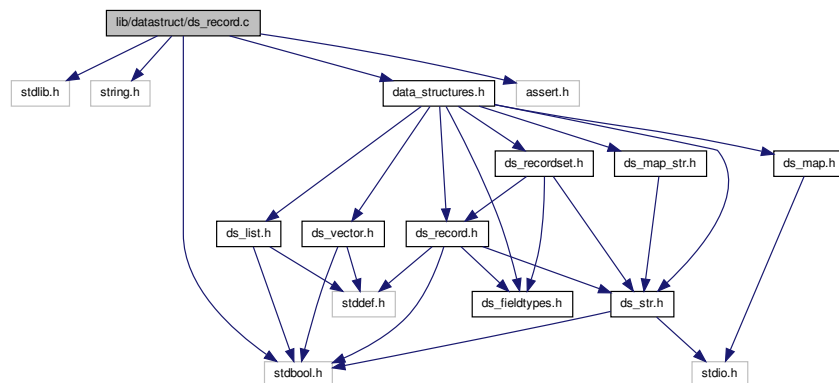
Parameters

<i>map</i>	A reference to the hash map.
<i>key</i>	The key.
<i>value</i>	The value.

5.64 lib/datastruct/ds_record.c File Reference

Implementation of record database structure.

```
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <assert.h>
#include "data_structures.h"
Include dependency graph for ds_record.c:
```



Data Structures

- struct [ds_record](#)

Functions

- [ds_record ds_record_create](#) (const size_t size)
Creates a new record.
- void [ds_record_destroy](#) (ds_record record)
Destroys a record and frees any associated resources.
- void [ds_record_destructor](#) (void *record)

- A record destructor function.*
- void `ds_record_clear` (`ds_record` record)
- Clears and `free()`s all the elements in a record.*
- void `ds_record_set_field` (`ds_record` record, const `size_t` index, `ds_str` field)
- Sets a field of a record.*
- `ds_str` `ds_record_get_field` (`ds_record` record, const `size_t` index)
- Retrieves the field at a specified index.*
- `size_t` `ds_record_size` (`ds_record` record)
- Returns the size of a record.*
- void `ds_record_seek_start` (`ds_record` record)
- Sets the current field to the first field of a record.*
- `ds_str` `ds_record_get_next_data` (`ds_record` record)
- Returns the next field of the record.*
- `ds_record` `ds_record_tokenize` (`ds_str` str, const char delim)
- Tokenizes a string into a record.*
- `ds_str` `ds_record_make_delim_string` (`ds_record` record, const char delim)
- Makes a delimited string from a record.*
- `ds_str` `ds_record_make_values_string` (`ds_record` record, enum `ds_field_types` *types)
- Makes a delimited SQL values string from a record.*

5.64.1 Detailed Description

Implementation of record database structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.64.2 Function Documentation

5.64.2.1 void ds_record_clear (ds_record record)

Clears and `free()`s all the elements in a record.

Parameters

<i>record</i>	The record.
---------------	-------------

5.64.2.2 ds_record ds_record_create (const size_t size)

Creates a new record.

Parameters

<i>size</i>	The size of the record.
-------------	-------------------------

Returns

A newly created record, or `NULL` on failure.

5.64.2.3 void ds_record_destroy (ds_record record)

Destroys a record and frees any associated resources.

Parameters

<i>record</i>	The record to destroy.
---------------	------------------------

5.64.2.4 void ds_record_destructor (void * record)

A record destructor function.

Parameters

<i>record</i>	The record to destroy.
---------------	------------------------

5.64.2.5 ds_str ds_record_get_field (ds_record record, const size_t index)

Retrieves the field at a specified index.

Parameters

<i>record</i>	The record from which to retrieve.
<i>index</i>	The index of the desired field.

Returns

A pointer to the field, or `NULL` if the index is out of range.

5.64.2.6 ds_str ds_record_get_next_data (ds_record record)

Returns the next field of the record.

This function returns the data of the "current field", and advances the current field pointer. Subsequent calls to this function will return successive fields.

Parameters

<i>record</i>	The record.
---------------	-------------

Returns

A pointer to the next field, or `NULL` if the end of the record has been reached.

5.64.2.7 ds_str ds_record_make_delim_string (ds_record record, const char delim)

Makes a delimited string from a record.

Parameters

<i>record</i>	The record.
<i>delim</i>	The delimiting character.

Returns

The delimited string, or `NULL` on failure.

5.64.2.8 `ds_str ds_record_make_values_string (ds_record record, enum ds_field_types * types)`

Makes a delimited SQL values string from a record.

Parameters

<i>record</i>	The record.
<i>types</i>	An array of types for each field, or <code>NULL</code> to assume they are all strings. The effect of this parameter is that string fields are quoted in the values string, whereas non-string fields are not.

Returns

The delimited values string, or `NULL` on failure.

5.64.2.9 `void ds_record_seek_start (ds_record record)`

Sets the current field to the first field of a record.

Parameters

<i>record</i>	The record.
---------------	-------------

5.64.2.10 `void ds_record_set_field (ds_record record, const size_t index, ds_str field)`

Sets a field of a record.

If the field is currently occupied, the existing field is `free()`d.

Parameters

<i>record</i>	The record to set.
<i>index</i>	The index of the field to set.
<i>field</i>	The value to which to set the field.

5.64.2.11 `size_t ds_record_size (ds_record record)`

Returns the size of a record.

Parameters

<i>record</i>	The record.
---------------	-------------

Returns

The size of the record.

5.64.2.12 ds_record ds_record.tokenize (ds_str str, const char delim)

Tokenizes a string into a record.

Parameters

<i>str</i>	The string to tokenize.
<i>delim</i>	The delimiting character.

Returns

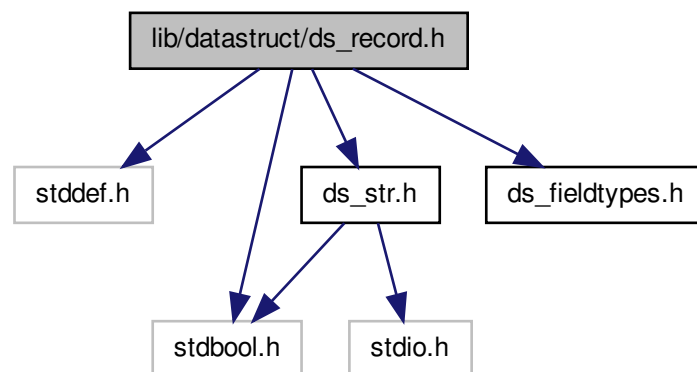
A new record containing the tokens.

5.65 lib/datastruct/ds_record.h File Reference

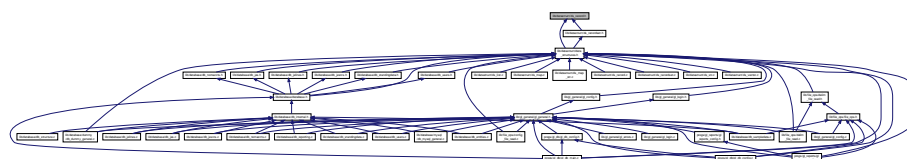
Interface to record data structure.

```
#include <stddef.h>
#include <stdbool.h>
#include "ds_str.h"
#include "ds_fieldtypes.h"
```

Include dependency graph for ds_record.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct [ds_record](#) * [ds_record](#)

Functions

- [ds_record ds_record_create](#) (const [size_t](#) size)
Creates a new record.
- void [ds_record_destroy](#) ([ds_record](#) record)
Destroys a record and frees any associated resources.
- void [ds_record_destructor](#) (void *record)
A record destructor function.
- void [ds_record_clear](#) ([ds_record](#) record)
Clears and `free()`s all the elements in a record.
- void [ds_record_set_field](#) ([ds_record](#) record, const [size_t](#) index, [ds_str](#) field)
Sets a field of a record.
- [ds_str ds_record_get_field](#) ([ds_record](#) record, const [size_t](#) index)
Retrieves the field at a specified index.
- [size_t ds_record_size](#) ([ds_record](#) record)
Returns the size of a record.
- void [ds_record_seek_start](#) ([ds_record](#) record)
Sets the current field to the first field of a record.
- [ds_str ds_record_get_next_data](#) ([ds_record](#) record)
Returns the next field of the record.
- [ds_record ds_record_tokenize](#) ([ds_str](#) str, const char delim)
Tokenizes a string into a record.
- [ds_str ds_record_make_delim_string](#) ([ds_record](#) record, const char delim)
Makes a delimited string from a record.
- [ds_str ds_record_make_values_string](#) ([ds_record](#) record, enum [ds_field_types](#) *types)
Makes a delimited SQL values string from a record.

5.65.1 Detailed Description

Interface to record data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.65.2 Typedef Documentation

5.65.2.1 typedef struct [ds_record](#)* [ds_record](#)

Typedef for opaque record datatype

5.65.3 Function Documentation

5.65.3.1 void ds_record_clear (ds_record record)

Clears and `free()`s all the elements in a record.

Parameters

<i>record</i>	The record.
---------------	-------------

5.65.3.2 ds_record ds_record_create (const size_t size)

Creates a new record.

Parameters

<i>size</i>	The size of the record.
-------------	-------------------------

Returns

A newly created record, or `NULL` on failure.

5.65.3.3 void ds_record_destroy (ds_record record)

Destroys a record and frees any associated resources.

Parameters

<i>record</i>	The record to destroy.
---------------	------------------------

5.65.3.4 void ds_record_destructor (void * record)

A record destructor function.

Parameters

<i>record</i>	The record to destroy.
---------------	------------------------

5.65.3.5 ds_str ds_record_get_field (ds_record record, const size_t index)

Retrieves the field at a specified index.

Parameters

<i>record</i>	The record from which to retrieve.
<i>index</i>	The index of the desired field.

Returns

A pointer to the field, or `NULL` if the index is out of range.

5.65.3.6 ds_str ds_record_get_next_data (ds_record record)

Returns the next field of the record.

This function returns the data of the "current field", and advances the current field pointer. Subsequent calls to this function will return successive fields.

Parameters

<i>record</i>	The record.
---------------	-------------

Returns

A pointer to the next field, or `NULL` if the end of the record has been reached.

5.65.3.7 ds_str ds_record_make_delim_string (ds_record record, const char delim)

Makes a delimited string from a record.

Parameters

<i>record</i>	The record.
<i>delim</i>	The delimiting character.

Returns

The delimited string, or `NULL` on failure.

5.65.3.8 ds_str ds_record_make_values_string (ds_record record, enum ds_field_types * types)

Makes a delimited SQL values string from a record.

Parameters

<i>record</i>	The record.
<i>types</i>	An array of types for each field, or <code>NULL</code> to assume they are all strings. The effect of this parameter is that string fields are quoted in the values string, whereas non-string fields are not.

Returns

The delimited values string, or `NULL` on failure.

5.65.3.9 void ds_record_seek_start (ds_record record)

Sets the current field to the first field of a record.

Parameters

<i>record</i>	The record.
---------------	-------------

5.65.3.10 void ds_record_set_field (ds_record record, const size_t index, ds_str field)

Sets a field of a record.

If the field is currently occupied, the existing field is `free()`d.

Parameters

<i>record</i>	The record to set.
<i>index</i>	The index of the field to set.
<i>field</i>	The value to which to set the field.

5.65.3.11 `size_t ds_record_size (ds_record record)`

Returns the size of a record.

Parameters

<i>record</i>	The record.
---------------	-------------

Returns

The size of the record.

5.65.3.12 `ds_record ds_record_tokenize (ds_str str, const char delim)`

Tokenizes a string into a record.

Parameters

<i>str</i>	The string to tokenize.
<i>delim</i>	The delimiting character.

Returns

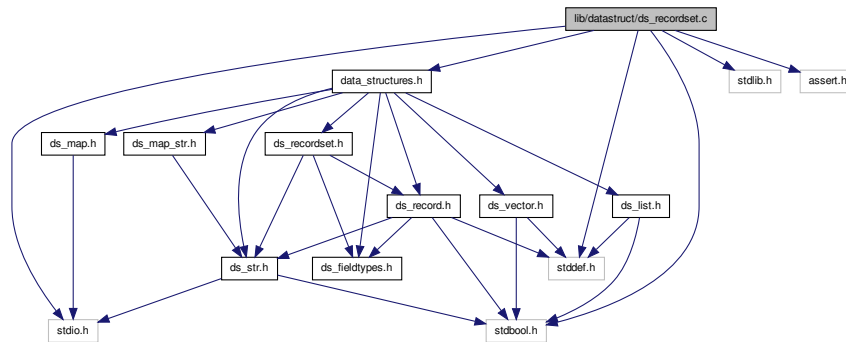
A new record containing the tokens.

5.66 `lib/datastruct/ds_recordset.c` File Reference

Implementation of query result set structure.

```
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <stdbool.h>
#include <assert.h>
#include "data_structures.h"
```

Include dependency graph for ds_recordset.c:



Data Structures

- struct [ds_recordset](#)

Functions

- [ds_recordset ds_recordset_create](#) (const [size_t](#) num_fields)
Creates a new record set.
- void [ds_recordset_destroy](#) ([ds_recordset](#) set)
Destroys a record set and frees associated resources.
- [ds_record](#) [ds_recordset_add_record](#) ([ds_recordset](#) set, [ds_record](#) record)
Adds a record to a record set.
- [size_t](#) [ds_recordset_num_fields](#) ([ds_recordset](#) set)
Returns the number of fields in a record set.
- [size_t](#) [ds_recordset_num_records](#) ([ds_recordset](#) set)
Returns the number of records in a record set.
- void [ds_recordset_set_headers](#) ([ds_recordset](#) set, [ds_record](#) headers)
Sets the record headers in a record set.
- void [ds_recordset_set_type](#) ([ds_recordset](#) set, const [size_t](#) index, const enum [ds_field_types](#) type)
Sets the type for a specified field.
- [ds_str](#) [ds_recordset_get_text_report](#) ([ds_recordset](#) set)
Returns a formatted text report for the record set.
- void [ds_recordset_seek_start](#) ([ds_recordset](#) set)
Sets the current record to the first record.
- [ds_record](#) [ds_recordset_next_record](#) ([ds_recordset](#) set)
Returns the next record in the record set.
- [ds_str](#) [ds_recordset_get_next_insert_query](#) ([ds_recordset](#) set, const char *table_name)
Gets the next SQL INSERT query.

5.66.1 Detailed Description

Implementation of query result set structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.66.2 Function Documentation

5.66.2.1 `ds_record ds_recordset_add_record (ds_recordset set, ds_record record)`

Adds a record to a record set.

The record *must* have the same number of fields as the number of fields provided to `ds_recordset_create()`.

Parameters

<i>set</i>	The record set to which to add.
<i>record</i>	The record to add.

Returns

A pointer to the new record (i.e. it returns the second parameter) or `NULL` on failure.

5.66.2.2 `ds_recordset ds_recordset_create (const size_t num_fields)`

Creates a new record set.

Parameters

<i>num_fields</i>	The non-zero number of fields in the record set.
-------------------	--

Returns

A pointer to the new record set.

5.66.2.3 `void ds_recordset_destroy (ds_recordset set)`

Destroys a record set and frees associated resources.

Parameters

<i>set</i>	The record set to destroy.
------------	----------------------------

5.66.2.4 `ds_str ds_recordset_get_next_insert_query (ds_recordset set, const char * table_name)`

Gets the next SQL INSERT query.

Parameters

<i>set</i>	The set.
<i>table_name</i>	The table name into which to insert.

Returns

The query. Caller is responsible for `free()` ing.

5.66.2.5 `ds_str ds_recordset_get_text_report (ds_recordset set)`

Returns a formatted text report for the record set.

The report is returned as a single multi-line string.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

A pointer to the report. The caller is responsible for `free()` ing this pointer.

5.66.2.6 `ds_record ds_recordset_next_record (ds_recordset set)`

Returns the next record in the record set.

This function returns the "current record", and advances the current record pointer. Subsequent calls to this function will return successive records.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

A pointer to the next record, or `NULL` if the end of the record set has been reached.

5.66.2.7 `size_t ds_recordset_num_fields (ds_recordset set)`

Returns the number of fields in a record set.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

The number of fields in the record set.

5.66.2.8 `size_t ds_recordset_num_records (ds_recordset set)`

Returns the number of records in a record set.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

The number of records in the record set.

5.66.2.9 void ds_recordset_seek_start (ds_recordset set)

Sets the current record to the first record.

Parameters

<i>set</i>	The record set.
------------	-----------------

5.66.2.10 void ds_recordset_set_headers (ds_recordset set, ds_record headers)

Sets the record headers in a record set.

Parameters

<i>set</i>	The record set.
<i>headers</i>	The headers, in the form of a ds_record of strings. The list <i>must</i> have the same number of elements as the number of fields provided to ds_recordset_create() .

5.66.2.11 void ds_recordset_set_type (ds_recordset set, const size_t index, const enum ds_field_types type)

Sets the type for a specified field.

Parameters

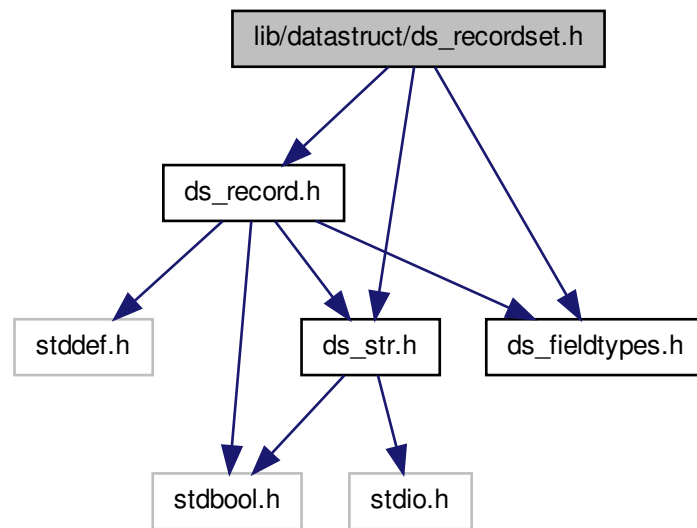
<i>set</i>	The record set.
<i>index</i>	The index to set.
<i>type</i>	The type for the field at the specified index.

5.67 lib/datastruct/ds_recordset.h File Reference

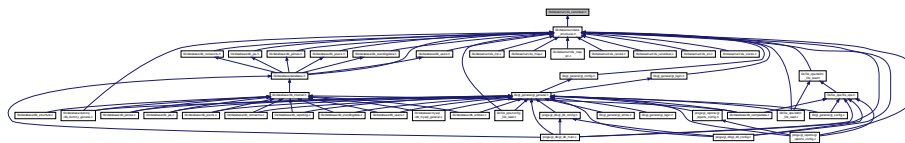
Interface to record set structure.

```
#include "ds_record.h"
#include "ds_str.h"
#include "ds_fieldtypes.h"
```


Include dependency graph for ds_recordset.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct `ds_recordset` * `ds_recordset`

Functions

- `ds_recordset` `ds_recordset_create` (const size_t num_fields)
Creates a new record set.
- void `ds_recordset_destroy` (`ds_recordset` set)
Destroys a record set and frees associated resources.
- `ds_record` `ds_recordset_add_record` (`ds_recordset` set, `ds_record` record)
Adds a record to a record set.
- size_t `ds_recordset_num_fields` (`ds_recordset` set)
Returns the number of fields in a record set.
- size_t `ds_recordset_num_records` (`ds_recordset` set)
Returns the number of records in a record set.
- void `ds_recordset_set_headers` (`ds_recordset` set, `ds_record` headers)
Sets the record headers in a record set.

- void `ds_recordset_set_type` (`ds_recordset` set, const size_t index, const enum `ds_field_types` type)
Sets the type for a specified field.
- `ds_str ds_recordset_get_text_report` (`ds_recordset` set)
Returns a formatted text report for the record set.
- `ds_str ds_recordset_get_next_insert_query` (`ds_recordset` set, const char *table_name)
Gets the next SQL INSERT query.
- void `ds_recordset_seek_start` (`ds_recordset` set)
Sets the current record to the first record.
- `ds_record ds_recordset_next_record` (`ds_recordset` set)
Returns the next record in the record set.

5.67.1 Detailed Description

Interface to record set structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.67.2 Typedef Documentation

5.67.2.1 typedef struct `ds_recordset*` `ds_recordset`

Typedef for opaque record set data type

5.67.3 Function Documentation

5.67.3.1 `ds_record ds_recordset_add_record` (`ds_recordset` set, `ds_record` record)

Adds a record to a record set.

The record *must* have the same number of fields as the number of fields provided to `ds_recordset_create()`.

Parameters

<i>set</i>	The record set to which to add.
<i>record</i>	The record to add.

Returns

A pointer to the new record (i.e. it returns the second parameter) or `NULL` on failure.

5.67.3.2 `ds_recordset ds_recordset_create` (const size_t num_fields)

Creates a new record set.

Parameters

<i>num_fields</i>	The non-zero number of fields in the record set.
-------------------	--

Returns

A pointer to the new record set.

5.67.3.3 void ds_recordset_destroy (ds_recordset set)

Destroys a record set and frees associated resources.

Parameters

<i>set</i>	The record set to destroy.
------------	----------------------------

5.67.3.4 ds_str ds_recordset_get_next_insert_query (ds_recordset set, const char * table_name)

Gets the next SQL INSERT query.

Parameters

<i>set</i>	The set.
<i>table_name</i>	The table name into which to insert.

Returns

The query. Caller is responsible for `free()` ing.

5.67.3.5 ds_str ds_recordset_get_text_report (ds_recordset set)

Returns a formatted text report for the record set.

The report is returned as a single multi-line string.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

A pointer to the report. The caller is responsible for `free()` ing this pointer.

5.67.3.6 ds_record ds_recordset_next_record (ds_recordset set)

Returns the next record in the record set.

This function returns the "current record", and advances the current record pointer. Subsequent calls to this function will return successive records.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

A pointer to the next record, or `NULL` if the end of the record set has been reached.

5.67.3.7 `size_t ds_recordset_num_fields (ds_recordset set)`

Returns the number of fields in a record set.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

The number of fields in the record set.

5.67.3.8 `size_t ds_recordset_num_records (ds_recordset set)`

Returns the number of records in a record set.

Parameters

<i>set</i>	The record set.
------------	-----------------

Returns

The number of records in the record set.

5.67.3.9 `void ds_recordset_seek_start (ds_recordset set)`

Sets the current record to the first record.

Parameters

<i>set</i>	The record set.
------------	-----------------

5.67.3.10 `void ds_recordset_set_headers (ds_recordset set, ds_record headers)`

Sets the record headers in a record set.

Parameters

<i>set</i>	The record set.
<i>headers</i>	The headers, in the form of a ds_record of strings. The list <i>must</i> have the same number of elements as the number of fields provided to ds_recordset_create() .

5.67.3.11 `void ds_recordset_set_type (ds_recordset set, const size_t index, const enum ds_field_types type)`

Sets the type for a specified field.

Parameters

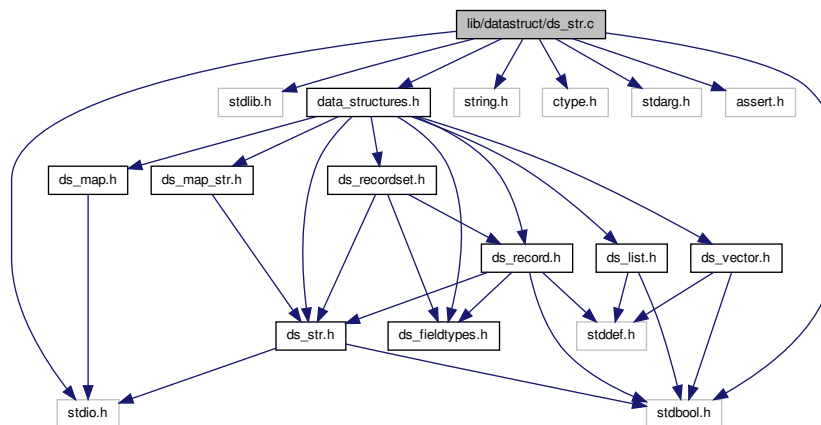
<i>set</i>	The record set.
<i>index</i>	The index to set.
<i>type</i>	The type for the field at the specified index.

5.68 lib/datastruct/ds_str.c File Reference

Implementation of string data structure.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <string.h>
#include <ctype.h>
#include <stdarg.h>
#include <assert.h>
#include "data_structures.h"
```

Include dependency graph for ds_str.c:



Data Structures

- struct [ds_str](#)

Functions

- [ds_str ds_str_create_direct](#) (char *init_str, const size_t init_str_size)
Creates a string using allocated memory.
- [ds_str ds_str_create](#) (const char *init_str)
Creates a new string from a C-style string.
- [ds_str ds_str_dup](#) (ds_str src)
Creates a new string from another string.
- [ds_str ds_str_create_sprintf](#) (const char *format,...)
Creates a string with `sprintf()`-type format.
- void [ds_str_destroy](#) (ds_str str)
Destroys a string and releases allocated resources.
- void [ds_str_destructor](#) (void *str)
Destroys a string and releases allocated resources.
- [ds_str ds_str_assign](#) (ds_str dst, ds_str src)
Assigns a string to another.
- [ds_str ds_str_assign_cstr](#) (ds_str dst, const char *src)
Assigns a C-style string to a string.

- `const char * ds_str_cstr (ds_str str)`
Returns a C-style string containing the string's contents.
- `size_t ds_str_length (ds_str str)`
Returns the length of a string.
- `ds_str ds_str_size_to_fit (ds_str str)`
Reduces a string's capacity to fit its length.
- `ds_str ds_str_concat (ds_str dst, ds_str src)`
Concatenates two strings.
- `ds_str ds_str_concat_cstr (ds_str dst, const char *src)`
Concatenates a C-style string to a string.
- `ds_str ds_str_trunc (ds_str str, const size_t length)`
Truncates a string.
- `unsigned long ds_str_hash (ds_str str)`
Calculates a hash of a string.
- `int ds_str_compare (ds_str s1, ds_str s2)`
Compares two strings.
- `int ds_str_compare_cstr (ds_str s1, const char *s2)`
Compares a string with a C-style string.
- `int ds_str_strchr (ds_str str, const char ch, const int start)`
Returns index of first occurrence of a character.
- `ds_str ds_str_substr_left (ds_str str, const size_t numchars)`
Returns a left substring.
- `ds_str ds_str_substr_right (ds_str str, const size_t numchars)`
Returns a right substring.
- `void ds_str_split (ds_str src, ds_str *left, ds_str *right, const char sc)`
Splits a string.
- `void ds_str_trim_leading (ds_str str)`
Trims leading whitespace in-place.
- `void ds_str_trim_trailing (ds_str str)`
Trims trailing whitespace in-place.
- `void ds_str_trim (ds_str str)`
Trims leading and trailing whitespace in-place.
- `char ds_str_char_at_index (ds_str str, const size_t index)`
Returns the character at a specified index.
- `bool ds_str_is_empty (ds_str str)`
Checks if a string is empty.
- `bool ds_str_is_alnum (ds_str str)`
Checks if a string contains only alphanumeric characters.
- `void ds_str_clear (ds_str str)`
Clears (empties) a string.
- `bool ds_str_intval (ds_str str, const int base, int *value)`
Gets the integer value of a string.
- `bool ds_str_doubleval (ds_str str, double *value)`
Gets the double value of a string.
- `ds_str ds_str_getline (ds_str str, const size_t size, FILE *fp)`
Gets a line from a file and assigns it to a string.
- `ds_str ds_str_decorate (ds_str str, ds_str left_dec, ds_str right_dec)`
Brackets a string with decoration strings.

5.68.1 Detailed Description

Implementation of string data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.68.2 Function Documentation

5.68.2.1 `ds_str ds_str_assign (ds_str dst, ds_str src)`

Assigns a string to another.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source string.

Returns

dst on success, `NULL` on failure.

5.68.2.2 `ds_str ds_str_assign_cstr (ds_str dst, const char * src)`

Assigns a C-style string to a string.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source C-style string.

Returns

dst on success, `NULL` on failure.

5.68.2.3 `char ds_str_char_at_index (ds_str str, const size_t index)`

Returns the character at a specified index.

Parameters

<i>str</i>	The string.
<i>index</i>	The specified index.

Returns

The character at the specified index.

5.68.2.4 void ds_str_clear (ds_str str)

Clears (empties) a string.

Parameters

<i>str</i>	The string.
------------	-------------

5.68.2.5 int ds_str_compare (ds_str s1, ds_str s2)

Compares two strings.

Parameters

<i>s1</i>	The first string.
<i>s2</i>	The second string.

Returns

Less than, equal to, or greater than zero if s1 is found, respectively, to be less than, equal to, or greater than s2.

5.68.2.6 int ds_str_compare_cstr (ds_str s1, const char * s2)

Compares a string with a C-style string.

Parameters

<i>s1</i>	The first string.
<i>s2</i>	The second, C-Style string.

Returns

Less than, equal to, or greater than zero if s1 is found, respectively, to be less than, equal to, or greater than s2.

5.68.2.7 ds_str ds_str_concat (ds_str dst, ds_str src)

Concatenates two strings.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source strings.

Returns

The destination string, or NULL on failure.

5.68.2.8 ds_str ds_str_concat_cstr (ds_str dst, const char * src)

Concatenates a C-style string to a string.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source strings.

Returns

The destination string, or `NULL` on failure.

5.68.2.9 ds_str ds_str_create (const char * *init_str*)

Creates a new string from a C-style string.

Parameters

<i>init_str</i>	The C-style string.
-----------------	---------------------

Returns

The new string, or `NULL` on failure.

5.68.2.10 ds_str ds_str_create_direct (char * *init_str*, const size_t *init_str_size*)

Creates a string using allocated memory.

The normal construction functions duplicate the string used to create it. In cases where allocated memory is already available (e.g. in `ds_str_create_sprintf()`) this function allows that memory to be directly assigned to the string, avoiding an unnecessary duplication.

Parameters

<i>init_str</i>	The allocated memory. IMPORTANT: If the construction of the string fails, this memory will be <code>free()</code> d.
<i>init_str_size</i>	The size of the allocated memory. IMPORTANT: The string's length is assumed to be one less than this quantity, and a call to <code>strlen()</code> is NOT performed.

Returns

The new string, or `NULL` on failure.

5.68.2.11 ds_str ds_str_create_sprintf (const char * *format*, ...)

Creates a string with `sprintf()`-type format.

Parameters

<i>format</i>	The format string.
...	The subsequent arguments as specified by the format string.

Returns

The new string, or `NULL` on failure.

5.68.2.12 `const char* ds_str_cstr (ds_str str)`

Returns a C-style string containing the string's contents.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

The C-style string containing the string's contents. The caller should not directly modify this string.

5.68.2.13 `ds_str ds_str_decorate (ds_str str, ds_str left_dec, ds_str right_dec)`

Brackets a string with decoration strings.

Parameters

<i>str</i>	The string to decorate.
<i>left_dec</i>	The string to add to the left of <i>str</i> .
<i>right_dec</i>	The string to add to the right of <i>str</i> , or NULL to add <i>left_dec</i> to both sides.

Returns

The decorated string.

5.68.2.14 `void ds_str_destroy (ds_str str)`

Destroys a string and releases allocated resources.

Parameters

<i>str</i>	The string to destroy..
------------	-------------------------

5.68.2.15 `void ds_str_destructor (void * str)`

Destroys a string and releases allocated resources.

This function calls `ds_str_destroy()`, and can be passed to a data structure expecting a destructor function with the signature `void (*)(void *)`.

Parameters

<i>str</i>	The string to destroy.
------------	------------------------

5.68.2.16 `bool ds_str_doubleval (ds_str str, double * value)`

Gets the double value of a string.

Parameters

<i>str</i>	The string.
<i>value</i>	A pointer to the double in which to store the value. Zero is stored if the string does not contain a valid double value.

Returns

`true` on successful conversion, `false` if the string does not contain a valid double value.

5.68.2.17 `ds_str ds_str_dup (ds_str src)`

Creates a new string from another string.

Parameters

<i>src</i>	The other string.
------------	-------------------

Returns

The new string, or `NULL` on failure.

5.68.2.18 `ds_str ds_str_getline (ds_str str, const size_t size, FILE * fp)`

Gets a line from a file and assigns it to a string.

Any trailing newline character is stripped.

Parameters

<i>str</i>	The string.
<i>size</i>	The maximum number of bytes to read, including the null.
<i>fp</i>	The file pointer from which to read.

Returns

`dst`

5.68.2.19 `unsigned long ds_str_hash (ds_str str)`

Calculates a hash of a string.

Uses Dan Bernstein's djb2 algorithm.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

The hash value

5.68.2.20 `bool ds_str_intval (ds_str str, const int base, int * value)`

Gets the integer value of a string.

Parameters

<i>str</i>	The string.
<i>base</i>	The base of the integer. This has the same meaning as the third argument to standard C <code>strtol()</code> .

<i>value</i>	A pointer to the integer in which to store the value. Zero is stored if the string does not contain a valid integer value.
--------------	--

Returns

`true` on successful conversion, `false` if the string does not contain a valid integer value.

5.68.2.21 `bool ds_str_is_alnum (ds_str str)`

Checks if a string contains only alphanumeric characters.

The string must contain *some* alphanumeric characters to check `true`, i.e. the string must be non-empty. Thus it can be used to check that a string does indeed contain content, and that that content is solely alphanumeric.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

`true` if the string contains only alphanumeric characters, `false` otherwise.

5.68.2.22 `bool ds_str_is_empty (ds_str str)`

Checks if a string is empty.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

`true` if the string is empty, `false` otherwise.

5.68.2.23 `size_t ds_str_length (ds_str str)`

Returns the length of a string.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

The length of the string.

5.68.2.24 `ds_str ds_str_size_to_fit (ds_str str)`

Reduces a string's capacity to fit its length.

Parameters

<i>str</i>	The string to size.
------------	---------------------

Returns

str, or NULL on failure.

5.68.2.25 void ds_str_split (ds_str *src*, ds_str * *left*, ds_str * *right*, const char *sc*)

Splits a string.

Parameters

<i>src</i>	The string to split.
<i>left</i>	Pointer to left substring (modified)
<i>right</i>	Pointer to right substring (modified)
<i>sc</i>	Split character.

5.68.2.26 int ds_str_strchr (ds_str *str*, const char *ch*, const int *start*)

Returns index of first occurrence of a character.

Parameters

<i>str</i>	The string.
<i>ch</i>	The character for which to search.
<i>start</i>	The index of the string at which to start looking. Set this to non-zero to begin searching from a point other than the first character of the string.

Returns

The index of the first occurrence, or -1 if the character was not found.

5.68.2.27 ds_str ds_str_substr_left (ds_str *str*, const size_t *numchars*)

Returns a left substring.

Parameters

<i>str</i>	The string.
<i>numchars</i>	The number of left characters to return. If this is greater than the length of the string, the whole string is returned.

Returns

A new string representing the substring.

5.68.2.28 ds_str ds_str_substr_right (ds_str *str*, const size_t *numchars*)

Returns a right substring.

Parameters

<i>str</i>	The string.
<i>numchars</i>	The number of right characters to return. If this is greater than the length of the string, the whole string is returned.

Returns

A new string representing the substring.

5.68.2.29 void ds_str_trim (ds_str str)

Trims leading and trailing whitespace in-place.

Parameters

<i>str</i>	The string.
------------	-------------

5.68.2.30 void ds_str_trim.leading (ds_str str)

Trims leading whitespace in-place.

Parameters

<i>str</i>	The string.
------------	-------------

5.68.2.31 void ds_str_trim.trailing (ds_str str)

Trims trailing whitespace in-place.

Parameters

<i>str</i>	The string.
------------	-------------

5.68.2.32 ds_str ds_str_trunc (ds_str str, const size_t length)

Truncates a string.

Parameters

<i>str</i>	The string.
<i>length</i>	The new length to which to truncate.

Returns

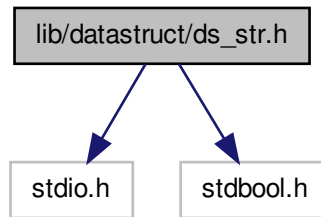
The original string, or `NULL` on failure.

5.69 lib/datastruct/ds_str.h File Reference

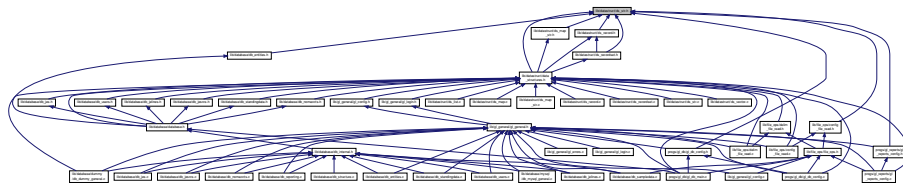
Interface to string data structure.

```
#include <stdio.h>
#include <stdbool.h>
```

Include dependency graph for ds_str.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct [ds_str](#) * [ds_str](#)

Functions

- [ds_str ds_str_create](#) (const char *init_str)
Creates a new string from a C-style string.
- [ds_str ds_str_dup](#) (ds_str src)
Creates a new string from another string.
- [ds_str ds_str_create_sprintf](#) (const char *format,...)
Creates a string with `sprintf()`-type format.
- [ds_str ds_str_create_direct](#) (char *init_str, const size_t init_str_size)
Creates a string using allocated memory.
- void [ds_str_destroy](#) (ds_str str)
Destroys a string and releases allocated resources.
- void [ds_str_destructor](#) (void *str)
Destroys a string and releases allocated resources.
- [ds_str ds_str_assign](#) (ds_str dst, ds_str src)
Assigns a string to another.
- [ds_str ds_str_assign_cstr](#) (ds_str dst, const char *src)
Assigns a C-style string to a string.
- const char * [ds_str_cstr](#) (ds_str str)
Returns a C-style string containing the string's contents.
- size_t [ds_str_length](#) (ds_str str)

- Returns the length of a string.*

 - `ds_str ds_str_size_to_fit (ds_str str)`

Reduces a string's capacity to fit its length.
- `ds_str ds_str_concat (ds_str dst, ds_str src)`

Concatenates two strings.
- `ds_str ds_str_concat_cstr (ds_str dst, const char *src)`

Concatenates a C-style string to a string.
- `ds_str ds_str_trunc (ds_str str, const size_t length)`

Truncates a string.
- unsigned long `ds_str_hash (ds_str str)`

Calculates a hash of a string.
- int `ds_str_compare (ds_str s1, ds_str s2)`

Compares two strings.
- int `ds_str_compare_cstr (ds_str s1, const char *s2)`

Compares a string with a C-style string.
- int `ds_str_strchr (ds_str str, const char ch, const int start)`

Returns index of first occurrence of a character.
- `ds_str ds_str_substr_left (ds_str str, const size_t numchars)`

Returns a left substring.
- `ds_str ds_str_substr_right (ds_str str, const size_t numchars)`

Returns a right substring.
- void `ds_str_split (ds_str src, ds_str *left, ds_str *right, const char sc)`

Splits a string.
- void `ds_str_trim_leading (ds_str str)`

Trims leading whitespace in-place.
- void `ds_str_trim_trailing (ds_str str)`

Trims trailing whitespace in-place.
- void `ds_str_trim (ds_str str)`

Trims leading and trailing whitespace in-place.
- char `ds_str_char_at_index (ds_str str, const size_t index)`

Returns the character at a specified index.
- bool `ds_str_is_empty (ds_str str)`

Checks if a string is empty.
- bool `ds_str_is_alnum (ds_str str)`

Checks if a string contains only alphanumeric characters.
- void `ds_str_clear (ds_str str)`

Clears (empties) a string.
- bool `ds_str_intval (ds_str str, const int base, int *value)`

Gets the integer value of a string.
- bool `ds_str_doubleval (ds_str str, double *value)`

Gets the double value of a string.
- `ds_str ds_str_getline (ds_str str, const size_t size, FILE *fp)`

Gets a line from a file and assigns it to a string.
- `ds_str ds_str_decorate (ds_str str, ds_str left_dec, ds_str right_dec)`

Brackets a string with decoration strings.

5.69.1 Detailed Description

Interface to string data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.69.2 Typedef Documentation

5.69.2.1 typedef struct ds_str* ds_str

Opaque data type for string

5.69.3 Function Documentation

5.69.3.1 ds_str ds_str_assign (ds_str dst, ds_str src)

Assigns a string to another.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source string.

Returns

dst on success, NULL on failure.

5.69.3.2 ds_str ds_str_assign_cstr (ds_str dst, const char * src)

Assigns a C-style string to a string.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source C-style string.

Returns

dst on success, NULL on failure.

5.69.3.3 char ds_str_char_at_index (ds_str str, const size_t index)

Returns the character at a specified index.

Parameters

<i>str</i>	The string.
<i>index</i>	The specified index.

Returns

The character at the specified index.

5.69.3.4 void ds_str_clear (ds_str str)

Clears (empties) a string.

Parameters

<i>str</i>	The string.
------------	-------------

5.69.3.5 int ds_str_compare (ds_str s1, ds_str s2)

Compares two strings.

Parameters

<i>s1</i>	The first string.
<i>s2</i>	The second string.

Returns

Less than, equal to, or greater than zero if *s1* is found, respectively, to be less than, equal to, or greater than *s2*.

5.69.3.6 int ds_str_compare_cstr (ds_str s1, const char * s2)

Compares a string with a C-style string.

Parameters

<i>s1</i>	The first string.
<i>s2</i>	The second, C-Style string.

Returns

Less than, equal to, or greater than zero if *s1* is found, respectively, to be less than, equal to, or greater than *s2*.

5.69.3.7 ds_str ds_str_concat (ds_str dst, ds_str src)

Concatenates two strings.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source strings.

Returns

The destination string, or `NULL` on failure.

5.69.3.8 ds_str ds_str_concat_cstr (ds_str dst, const char * src)

Concatenates a C-style string to a string.

Parameters

<i>dst</i>	The destination string.
<i>src</i>	The source strings.

Returns

The destination string, or `NULL` on failure.

5.69.3.9 ds_str ds_str_create (const char * init_str)

Creates a new string from a C-style string.

Parameters

<i>init_str</i>	The C-style string.
-----------------	---------------------

Returns

The new string, or `NULL` on failure.

5.69.3.10 ds_str ds_str_create_direct (char * init_str, const size_t init_str_size)

Creates a string using allocated memory.

The normal construction functions duplicate the string used to create it. In cases where allocated memory is already available (e.g. in `ds_str_create_sprintf()`) this function allows that memory to be directly assigned to the string, avoiding an unnecessary duplication.

Parameters

<i>init_str</i>	The allocated memory. IMPORTANT: If the construction of the string fails, this memory will be <code>free()</code> d.
<i>init_str_size</i>	The size of the allocated memory. IMPORTANT: The string's length is assumed to be one less than this quantity, and a call to <code>strlen()</code> is NOT performed.

Returns

The new string, or `NULL` on failure.

5.69.3.11 ds_str ds_str_create_sprintf (const char * format, ...)

Creates a string with `sprintf()`-type format.

Parameters

<i>format</i>	The format string.
<i>...</i>	The subsequent arguments as specified by the format string.

Returns

The new string, or `NULL` on failure.

5.69.3.12 `const char* ds_str_cstr (ds_str str)`

Returns a C-style string containing the string's contents.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

The C-style string containing the string's contents. The caller should not directly modify this string.

5.69.3.13 `ds_str ds_str_decorate (ds_str str, ds_str left_dec, ds_str right_dec)`

Brackets a string with decoration strings.

Parameters

<i>str</i>	The string to decorate.
<i>left_dec</i>	The string to add to the left of <i>str</i> .
<i>right_dec</i>	The string to add to the right of <i>str</i> , or <code>NULL</code> to add <i>left_dec</i> to both sides.

Returns

The decorated string.

5.69.3.14 `void ds_str_destroy (ds_str str)`

Destroys a string and releases allocated resources.

Parameters

<i>str</i>	The string to destroy..
------------	-------------------------

5.69.3.15 `void ds_str_destructor (void * str)`

Destroys a string and releases allocated resources.

This function calls `ds_str_destroy()`, and can be passed to a data structure expecting a destructor function with the signature `void (*)(void *)`.

Parameters

<i>str</i>	The string to destroy.
------------	------------------------

5.69.3.16 `bool ds_str_doubleval (ds_str str, double * value)`

Gets the double value of a string.

Parameters

<i>str</i>	The string.
<i>value</i>	A pointer to the double in which to store the value. Zero is stored if the string does not contain a valid double value.

Returns

`true` on successful conversion, `false` if the string does not contain a valid double value.

5.69.3.17 ds_str ds_str_dup (ds_str src)

Creates a new string from another string.

Parameters

<i>src</i>	The other string.
------------	-------------------

Returns

The new string, or `NULL` on failure.

5.69.3.18 ds_str ds_str_getline (ds_str str, const size_t size, FILE * fp)

Gets a line from a file and assigns it to a string.

Any trailing newline character is stripped.

Parameters

<i>str</i>	The string.
<i>size</i>	The maximum number of bytes to read, including the null.
<i>fp</i>	The file pointer from which to read.

Returns

`dst`

5.69.3.19 unsigned long ds_str_hash (ds_str str)

Calculates a hash of a string.

Uses Dan Bernstein's djb2 algorithm.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

The hash value

5.69.3.20 bool ds_str_intval (ds_str str, const int base, int * value)

Gets the integer value of a string.

Parameters

<i>str</i>	The string.
<i>base</i>	The base of the integer. This has the same meaning as the third argument to standard C <code>strtol()</code> .
<i>value</i>	A pointer to the integer in which to store the value. Zero is stored if the string does not contain a valid integer value.

Returns

`true` on successful conversion, `false` if the string does not contain a valid integer value.

5.69.3.21 `bool ds_str_is_alnum (ds_str str)`

Checks if a string contains only alphanumeric characters.

The string must contain *some* alphanumeric characters to check `true`, i.e. the string must be non-empty. Thus it can be used to check that a string does indeed contain content, and that that content is solely alphanumeric.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

`true` if the string contains only alphanumeric characters, `false` otherwise.

5.69.3.22 `bool ds_str_is_empty (ds_str str)`

Checks if a string is empty.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

`true` if the string is empty, `false` otherwise.

5.69.3.23 `size_t ds_str_length (ds_str str)`

Returns the length of a string.

Parameters

<i>str</i>	The string.
------------	-------------

Returns

The length of the string.

5.69.3.24 `ds_str ds_str_size_to_fit (ds_str str)`

Reduces a string's capacity to fit its length.

Parameters

<i>str</i>	The string to size.
------------	---------------------

Returns

str, or NULL on failure.

5.69.3.25 void ds_str_split (ds_str *src*, ds_str * *left*, ds_str * *right*, const char *sc*)

Splits a string.

Parameters

<i>src</i>	The string to split.
<i>left</i>	Pointer to left substring (modified)
<i>right</i>	Pointer to right substring (modified)
<i>sc</i>	Split character.

5.69.3.26 int ds_str_strchr (ds_str *str*, const char *ch*, const int *start*)

Returns index of first occurrence of a character.

Parameters

<i>str</i>	The string.
<i>ch</i>	The character for which to search.
<i>start</i>	The index of the string at which to start looking. Set this to non-zero to begin searching from a point other than the first character of the string.

Returns

The index of the first occurrence, or -1 if the character was not found.

5.69.3.27 ds_str ds_str_substr_left (ds_str *str*, const size_t *numchars*)

Returns a left substring.

Parameters

<i>str</i>	The string.
<i>numchars</i>	The number of left characters to return. If this is greater than the length of the string, the whole string is returned.

Returns

A new string representing the substring.

5.69.3.28 ds_str ds_str_substr_right (ds_str *str*, const size_t *numchars*)

Returns a right substring.

Parameters

<i>str</i>	The string.
<i>numchars</i>	The number of right characters to return. If this is greater than the length of the string, the whole string is returned.

Returns

A new string representing the substring.

5.69.3.29 void ds_str_trim (ds_str str)

Trims leading and trailing whitespace in-place.

Parameters

<i>str</i>	The string.
------------	-------------

5.69.3.30 void ds_str_trim.leading (ds_str str)

Trims leading whitespace in-place.

Parameters

<i>str</i>	The string.
------------	-------------

5.69.3.31 void ds_str_trim.trailing (ds_str str)

Trims trailing whitespace in-place.

Parameters

<i>str</i>	The string.
------------	-------------

5.69.3.32 ds_str ds_str_trunc (ds_str str, const size_t length)

Truncates a string.

Parameters

<i>str</i>	The string.
<i>length</i>	The new length to which to truncate.

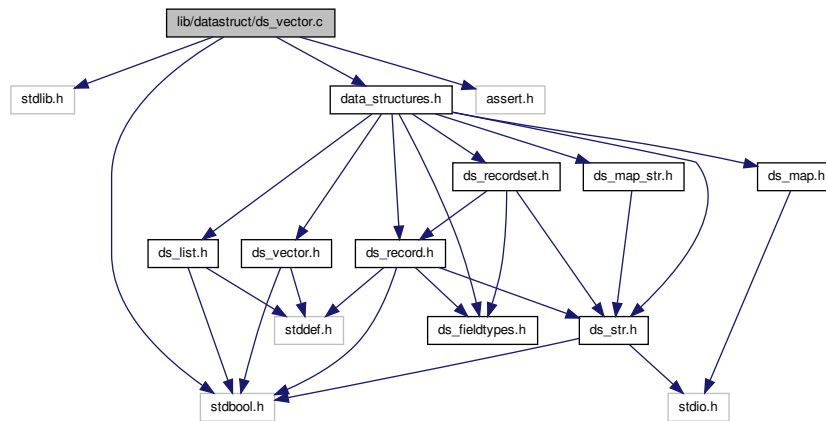
Returns

The original string, or `NULL` on failure.

5.70 lib/datastruct/ds_vector.c File Reference

Implementation of generic doubly-linked vector data structure.


```
#include <stdlib.h>
#include <stdbool.h>
#include <assert.h>
#include "data_structures.h"
Include dependency graph for ds_vector.c:
```



Data Structures

- struct [ds_vector](#)

Functions

- [ds_vector ds_vector_create](#) (const size_t size, const bool free_on_delete, void(*destructor)(void *))
Creates a new vector.
- void [ds_vector_destroy](#) (ds_vector vector)
Destroys a vector and frees any associated resources.
- void [ds_vector_destructor](#) (void *vector)
A vector destructor function.
- void [ds_vector_clear](#) (ds_vector vector)
Clears all the elements in a vector.
- void [ds_vector_set](#) (ds_vector vector, const size_t index, void *element)
Sets an element of a vector.
- void * [ds_vector_element](#) (ds_vector vector, const size_t index)
Retrieves the data at a specified index.
- size_t [ds_vector_size](#) (ds_vector vector)
Returns the size of a vector.
- void [ds_vector_seek_start](#) (ds_vector vector)
Sets the current element to the first element of a vector.
- void * [ds_vector_get_next_data](#) (ds_vector vector)
Returns the next element of the vector.

5.70.1 Detailed Description

Implementation of generic doubly-linked vector data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.70.2 Function Documentation**5.70.2.1 void ds_vector_clear (ds_vector vector)**

Clears all the elements in a vector.

If the vector was created with `free_on_delete`, the elements are `free()`d prior to being cleared (i.e. set to `NULL`).

Parameters

<i>vector</i>	The vector.
---------------	-------------

5.70.2.2 ds_vector ds_vector_create (const size_t size, const bool free_on_delete, void(*) (void *) destructor)

Creates a new vector.

Parameters

<i>size</i>	The size of the vector.
<i>free_on_delete</i>	Set to <code>true</code> if the vector elements should be destroyed when removed from the vector, and when the vector itself is destroyed. If set to <code>false</code> , the caller is responsible for destroying the elements prior to destroying the vector.
<i>destructor</i>	Pointer to a destructor function to use for destroying the vector elements, when <code>free_on_delete</code> is true. If this is set to <code>NULL</code> , <code>free()</code> from the standard C library will be used to destroy the elements.

Returns

A newly created vector, or `NULL` on failure.

5.70.2.3 void ds_vector_destroy (ds_vector vector)

Destroys a vector and frees any associated resources.

Parameters

<i>vector</i>	The vector to destroy.
---------------	------------------------

5.70.2.4 void ds_vector_destructor (void * vector)

A vector destructor function.

This function may be passed to `ds_vector_create()` when creating a vector of vectors. It calls `ds_vector_destroy()`, but the parameter of `ds_vector_destroy()` is not compatible with the function signature expected by `ds_vector_create()`, so this function provides an appropriate interface.

Parameters

<i>vector</i>	The vector to destroy.
---------------	------------------------

5.70.2.5 void* ds_vector_element (ds_vector vector, const size_t index)

Retrieves the data at a specified index.

Parameters

<i>vector</i>	The vector from which to retrieve.
<i>index</i>	The index of the desired element.

Returns

A pointer to the data, or `NULL` if the index is out of range.

5.70.2.6 void* ds_vector_get_next_data (ds_vector vector)

Returns the next element of the vector.

This function returns the data of the "current element", and advances the current element pointer. Subsequent calls to this function will return successive elements.

Parameters

<i>vector</i>	The vector.
---------------	-------------

Returns

A pointer to the next element, or `NULL` if the end of the vector has been reached.

5.70.2.7 void ds_vector_seek_start (ds_vector vector)

Sets the current element to the first element of a vector.

Parameters

<i>vector</i>	The vector.
---------------	-------------

5.70.2.8 void ds_vector_set (ds_vector vector, const size_t index, void * element)

Sets an element of a vector.

If the element is currently occupied, the existing element is `free()`d.

Parameters

<i>vector</i>	The vector to which to set.
<i>index</i>	The index of the element to set.
<i>element</i>	The element to set.

5.70.2.9 `size_t ds_vector_size (ds_vector vector)`

Returns the size of a vector.

Parameters

<i>vector</i>	The vector.
---------------	-------------

Returns

The size of the vector.

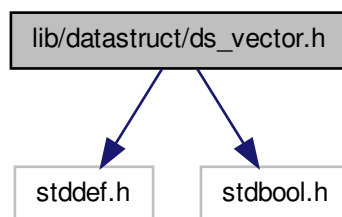
5.71 `lib/datastruct/ds_vector.h` File Reference

Interface to generic doubly-linked vector data structure.

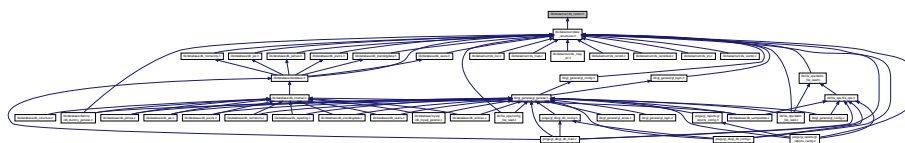
```
#include <stddef.h>
```

```
#include <stdbool.h>
```

Include dependency graph for `ds_vector.h`:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef struct [ds_vector](#) * [ds_vector](#)

Functions

- [ds_vector ds_vector_create](#) (const [size_t](#) size, const bool free_on_delete, void(*destructor)(void *))
Creates a new vector.
- void [ds_vector_destroy](#) ([ds_vector](#) vector)
Destroys a vector and frees any associated resources.

- void `ds_vector_destructor` (void *vector)
A vector destructor function.
- void `ds_vector_clear` (ds_vector vector)
Clears all the elements in a vector.
- void `ds_vector_set` (ds_vector vector, const size_t index, void *element)
Sets an element of a vector.
- void * `ds_vector_element` (ds_vector vector, const size_t index)
Retrieves the data at a specified index.
- size_t `ds_vector_size` (ds_vector vector)
Returns the size of a vector.
- void `ds_vector_seek_start` (ds_vector vector)
Sets the current element to the first element of a vector.
- void * `ds_vector_get_next_data` (ds_vector vector)
Returns the next element of the vector.

5.71.1 Detailed Description

Interface to generic doubly-linked vector data structure.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.71.2 Typedef Documentation

5.71.2.1 typedef struct ds_vector* ds_vector

Typedef for opaque vector datatype

5.71.3 Function Documentation

5.71.3.1 void ds_vector_clear (ds_vector vector)

Clears all the elements in a vector.

If the vector was created with `free_on_delete`, the elements are `free()`d prior to being cleared (i.e. set to NULL).

Parameters

<i>vector</i>	The vector.
---------------	-------------

5.71.3.2 ds_vector ds_vector_create (const size_t size, const bool free_on_delete, void(*) (void *) destructor)

Creates a new vector.

Parameters

<i>size</i>	The size of the vector.
<i>free_on_delete</i>	Set to <code>true</code> if the vector elements should be destroyed when removed from the vector, and when the vector itself is destroyed. If set to <code>false</code> , the caller is responsible for destroying the elements prior to destroying the vector.
<i>destructor</i>	Pointer to a destructor function to use for destroying the vector elements, when <code>free_on_delete</code> is true. If this is set to <code>NULL</code> , <code>free()</code> from the standard C library will be used to destroy the elements.

Returns

A newly created vector, or `NULL` on failure.

5.71.3.3 `void ds_vector_destroy (ds_vector vector)`

Destroys a vector and frees any associated resources.

Parameters

<i>vector</i>	The vector to destroy.
---------------	------------------------

5.71.3.4 `void ds_vector_destructor (void * vector)`

A vector destructor function.

This function may be passed to `ds_vector_create()` when creating a vector of vectors. It calls `ds_vector_destroy()`, but the parameter of `ds_vector_destroy()` is not compatible with the function signature expected by `ds_vector_create()`, so this function provides an appropriate interface.

Parameters

<i>vector</i>	The vector to destroy.
---------------	------------------------

5.71.3.5 `void* ds_vector_element (ds_vector vector, const size_t index)`

Retrieves the data at a specified index.

Parameters

<i>vector</i>	The vector from which to retrieve.
<i>index</i>	The index of the desired element.

Returns

A pointer to the data, or `NULL` if the index is out of range.

5.71.3.6 `void* ds_vector_get_next_data (ds_vector vector)`

Returns the next element of the vector.

This function returns the data of the "current element", and advances the current element pointer. Subsequent calls to this function will return successive elements.

Parameters

<i>vector</i>	The vector.
---------------	-------------

Returns

A pointer to the next element, or `NULL` if the end of the vector has been reached.

5.71.3.7 void ds_vector_seek_start (ds_vector vector)

Sets the current element to the first element of a vector.

Parameters

<i>vector</i>	The vector.
---------------	-------------

5.71.3.8 void ds_vector_set (ds_vector vector, const size_t index, void * element)

Sets an element of a vector.

If the element is currently occupied, the existing element is `free()`d.

Parameters

<i>vector</i>	The vector to which to set.
<i>index</i>	The index of the element to set.
<i>element</i>	The element to set.

5.71.3.9 size_t ds_vector_size (ds_vector vector)

Returns the size of a vector.

Parameters

<i>vector</i>	The vector.
---------------	-------------

Returns

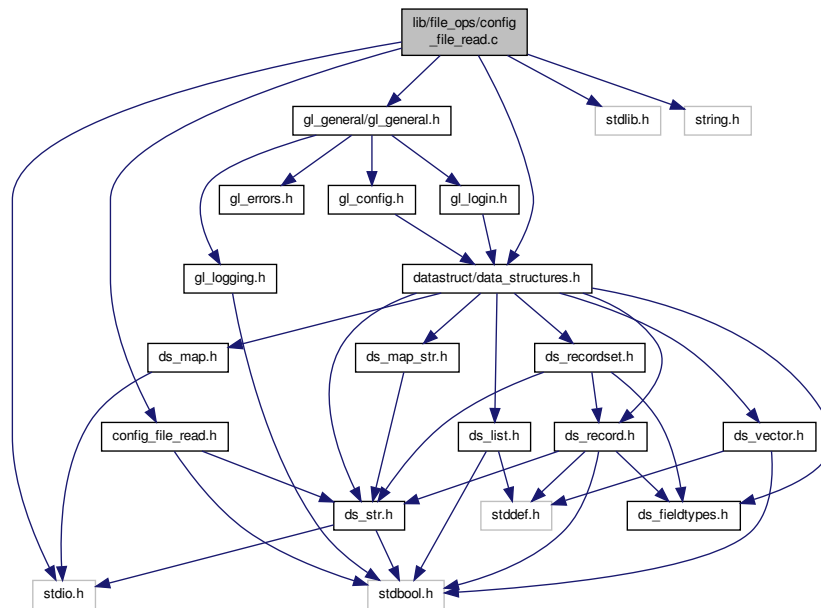
The size of the vector.

5.72 lib/file_ops/config_file_read.c File Reference

Implementation of configuration file reading functionality.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "gl_general/gl_general.h"
#include "datastruct/data_structures.h"
#include "config_file_read.h"
```

Include dependency graph for `config_file_read.c`:



Macros

- `#define MAX_BUFFER_SIZE 1024`
- `#define CONFIG_MAP_SIZE 100`

Functions

- `bool config_init (void)`
Initializes configuration data.
- `int config_file_read (const char *filename)`
Reads a configuration file and stores the key-value pairs.
- `ds_str config_value_get (ds_str key)`
Returns the value associated with a key.
- `ds_str config_value_get_cstr (const char *key)`
Returns the value associated with a C-style string key.
- `void config_value_set (ds_str key, ds_str value)`
Sets a key-value in the configuration structure.
- `void config_free (void)`
Frees the resources used by this module.

5.72.1 Detailed Description

Implementation of configuration file reading functionality. This module reads configuration files in the format "key = value" and makes those values available. Leading and trailing whitespace is removed for both the key and the value. Blank lines and lines starting with a '#' are ignored in the configuration file.

Author

Paul Griffiths

Copyright

Copyright 2013 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.72.2 Macro Definition Documentation**5.72.2.1 #define CONFIG_MAP_SIZE 100**

Size to use for the hash map to contain the key-value pairs

5.72.2.2 #define MAX_BUFFER_SIZE 1024

Maximum size of buffers

5.72.3 Function Documentation**5.72.3.1 int config_file_read (const char * *filename*)**

Reads a configuration file and stores the key-value pairs.

Parameters

<i>filename</i>	The name of the configuration file.
-----------------	-------------------------------------

Returns

CONFIG_FILE_OK on success, CONFIG_FILE_NO_FILE if the specified file could not be opened for reading, CONFIG_FILE_MALFORMED_FILE if the configuration file was improperly formed.

5.72.3.2 void config_free (void)

Frees the resources used by this module.

The user should make copies of any required keys or values prior to calling this function. This function need not be called if [config_file_read\(\)](#) returned an error.

5.72.3.3 bool config_init (void)

Initializes configuration data.

Returns

`true` on success, `false` on failure.

5.72.3.4 ds_str config_value_get (ds_str *key*)

Returns the value associated with a key.

Parameters

<i>key</i>	The specified key.
------------	--------------------

Returns

A pointer to the associated value, or `NULL` if the key was not present in the configuration file. The caller should not modify the string to which the pointer points.

5.72.3.5 `ds_str config_value_get_cstr (const char * key)`

Returns the value associated with a C-style string key.

Parameters

<i>key</i>	The specified key.
------------	--------------------

Returns

A pointer to the associated value, or `NULL` if the key was not present in the configuration file. The caller should not modify the string to which the pointer points.

5.72.3.6 `void config_value_set (ds_str key, ds_str value)`

Sets a key-value in the configuration structure.

Parameters

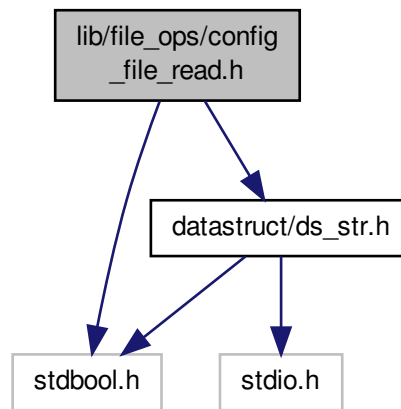
<i>key</i>	The key.
<i>value</i>	The value.

5.73 `lib/file_ops/config_file_read.h` File Reference

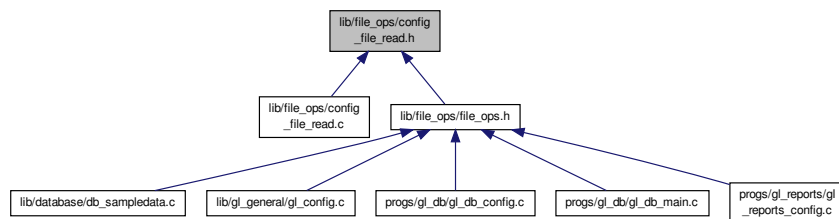
Interface to configuration file reading functionality.

```
#include <stdbool.h>
#include "datastruct/ds_str.h"
```

Include dependency graph for config_file_read.h:



This graph shows which files directly or indirectly include this file:



Macros

- `#define CONFIG_FILE_OK 0`
- `#define CONFIG_FILE_NO_FILE 1`
- `#define CONFIG_FILE_MALFORMED_FILE 2`

Functions

- `bool config_init (void)`
Initializes configuration data.
- `int config_file_read (const char *filename)`
Reads a configuration file and stores the key-value pairs.
- `void config_free (void)`
Frees the resources used by this module.
- `ds_str config_value_get (ds_str key)`
Returns the value associated with a key.
- `ds_str config_value_get_cstr (const char *key)`
Returns the value associated with a C-style string key.

- void `config_value_set` (`ds_str` key, `ds_str` value)
Sets a key-value in the configuration structure.

5.73.1 Detailed Description

Interface to configuration file reading functionality. This module reads configuration files in the format "key = value" and makes those values available. Leading and trailing whitespace is removed for both the key and the value. Blank lines and lines starting with a '#' are ignored in the configuration file.

Author

Paul Griffiths

Copyright

Copyright 2013 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.73.2 Macro Definition Documentation

5.73.2.1 #define CONFIG_FILE_MALFORMED_FILE 2

Return status when configuration file is improperly formed

5.73.2.2 #define CONFIG_FILE_NO_FILE 1

Return status when unable to open file for reading

5.73.2.3 #define CONFIG_FILE_OK 0

Return status for success

5.73.3 Function Documentation

5.73.3.1 int config_file_read (const char * *filename*)

Reads a configuration file and stores the key-value pairs.

Parameters

<i>filename</i>	The name of the configuration file.
-----------------	-------------------------------------

Returns

CONFIG_FILE_OK on success, CONFIG_FILE_NO_FILE if the specified file could not be opened for reading, CONFIG_FILE_MALFORMED_FILE if the configuration file was improperly formed.

5.73.3.2 void config_free (void)

Frees the resources used by this module.

The user should make copies of any required keys or values prior to calling this function. This function need not be called if `config_file_read()` returned an error.

5.73.3.3 `bool config_init (void)`

Initializes configuration data.

Returns

`true` on success, `false` on failure.

5.73.3.4 `ds_str config_value_get (ds_str key)`

Returns the value associated with a key.

Parameters

<i>key</i>	The specified key.
------------	--------------------

Returns

A pointer to the associated value, or `NULL` if the key was not present in the configuration file. The caller should not modify the string to which the pointer points.

5.73.3.5 `ds_str config_value_get_cstr (const char * key)`

Returns the value associated with a C-style string key.

Parameters

<i>key</i>	The specified key.
------------	--------------------

Returns

A pointer to the associated value, or `NULL` if the key was not present in the configuration file. The caller should not modify the string to which the pointer points.

5.73.3.6 `void config_value_set (ds_str key, ds_str value)`

Sets a key-value in the configuration structure.

Parameters

<i>key</i>	The key.
<i>value</i>	The value.

5.74 `lib/file_ops/delim_file_read.c` File Reference

Implementation of delimited file reading functionality.

5.74.2 Macro Definition Documentation

5.74.2.1 #define MAX_LINE_SIZE 1024

Maximum size of buffers

5.74.3 Function Documentation

5.74.3.1 ds_recordset delim_file_read (const char * filename, const char delim)

Constructs a [ds_recordset](#) from a delimited file.

Parameters

<i>filename</i>	The name of the delimited file.
<i>delim</i>	The delimiting character.

Returns

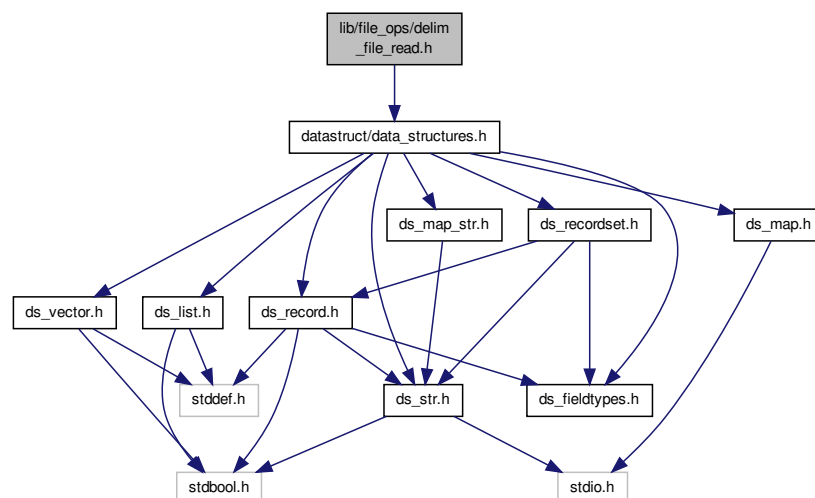
The [ds_recordset](#), or NULL on failure.

5.75 lib/file_ops/delim_file_read.h File Reference

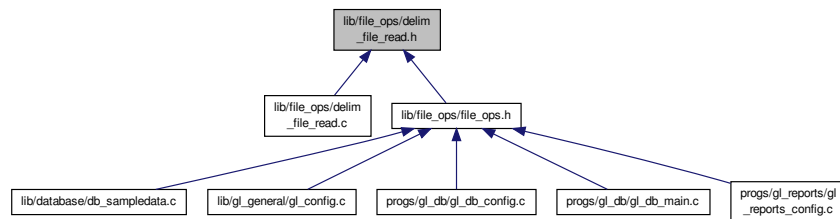
Interface to delimited file reading functionality.

```
#include "datastruct/data_structures.h"
```

Include dependency graph for `delim_file_read.h`:



This graph shows which files directly or indirectly include this file:



Functions

- [ds_recordset delim_file_read](#) (const char *filename, const char delim)
Constructs a [ds_recordset](#) from a delimited file.

5.75.1 Detailed Description

Interface to delimited file reading functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.75.2 Function Documentation

5.75.2.1 ds_recordset delim_file_read (const char * filename, const char delim)

Constructs a [ds_recordset](#) from a delimited file.

Parameters

<i>filename</i>	The name of the delimited file.
<i>delim</i>	The delimiting character.

Returns

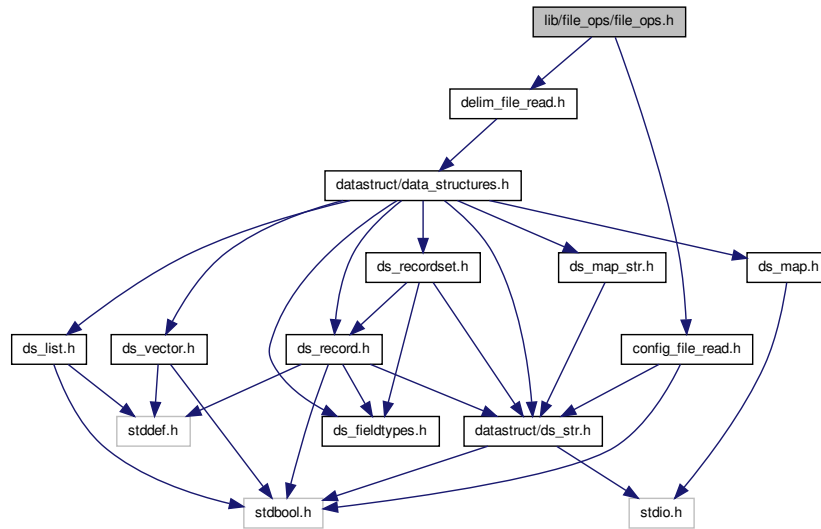
The [ds_recordset](#), or NULL on failure.

5.76 lib/file_ops/file_ops.h File Reference

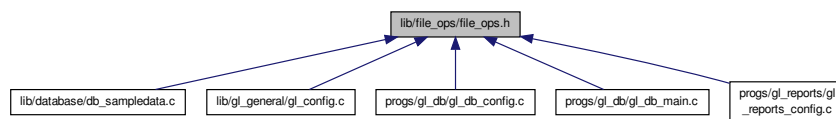
User interface to file operations functionality.

```
#include "config_file_read.h"
#include "delim_file_read.h"
```


Include dependency graph for file_ops.h:



This graph shows which files directly or indirectly include this file:



5.76.1 Detailed Description

User interface to file operations functionality.

Author

Paul Griffiths

Copyright

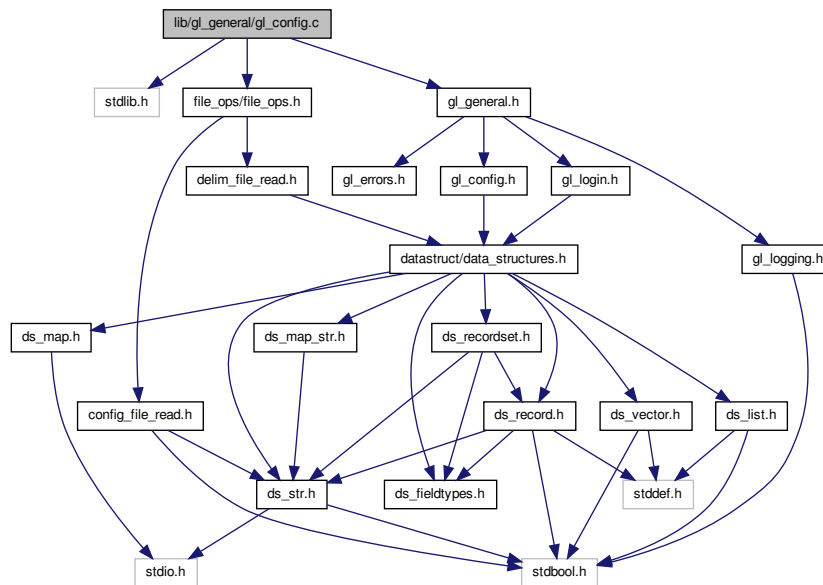
Copyright 2013 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.77 lib/gl_general/gl_config.c File Reference

Implementation of configuration functionality.

```
#include <stdlib.h>
#include "gl_general.h"
#include "file_ops/file_ops.h"
```

Include dependency graph for `gl_config.c`:



Functions

- struct `params` * `params_init` (void)
Initializes a parameters structure.
- void `params_free` (struct `params` *`params`)
Frees a parameter structure.
- bool `get_configuration` (struct `params` *`params`, const char *`conf_file`)
Gets parameters from a configuration file.

5.77.1 Detailed Description

Implementation of configuration functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.77.2 Function Documentation

5.77.2.1 bool `get_configuration` (struct `params` * `params`, const char * `conf_file`)

Gets parameters from a configuration file.

Parameters

<i>params</i>	A pointer to a parameters structure to populate.
<i>conf_file</i>	The filename of the configuration file.

Returns

`true` on success, `false` otherwise.

5.77.2.2 void params_free (struct params * params)

Frees a parameter structure.

Parameters

<i>params</i>	A pointer to the structure to free.
---------------	-------------------------------------

5.77.2.3 struct params* params_init (void) [read]

Initializes a parameters structure.

Returns

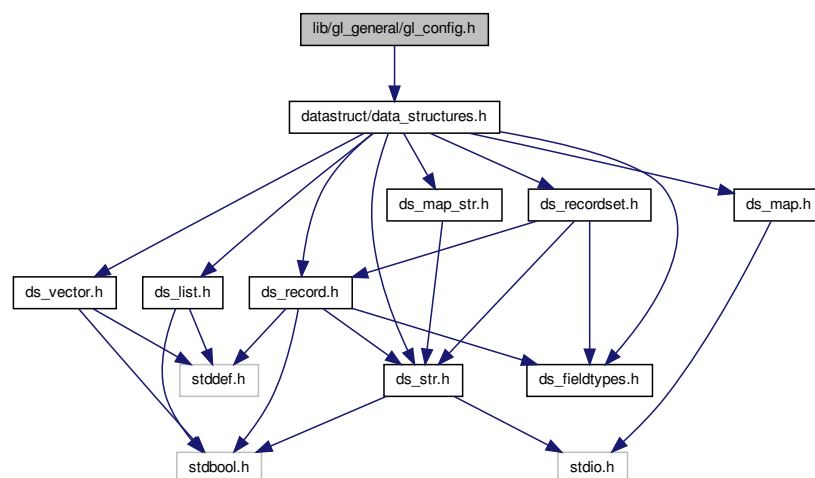
An initialized parameters structure.

5.78 lib/gl_general/gl_config.h File Reference

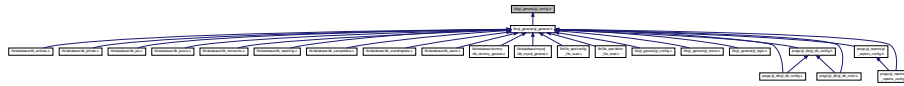
Interface to configuration functionality.

```
#include "datastruct/data_structures.h"
```

Include dependency graph for gl_config.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [params](#)

Functions

- struct [params](#) * [params_init](#) (void)
Initializes a parameters structure.
- void [params_free](#) (struct [params](#) *[params](#))
Frees a parameter structure.
- bool [get_configuration](#) (struct [params](#) *[params](#), const char *[conf_file](#))
Gets parameters from a configuration file.

5.78.1 Detailed Description

Interface to configuration functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.78.2 Function Documentation

5.78.2.1 bool [get_configuration](#) (struct [params](#) * [params](#), const char * [conf_file](#))

Gets parameters from a configuration file.

Parameters

<i>params</i>	A pointer to a parameters structure to populate.
<i>conf_file</i>	The filename of the configuration file.

Returns

`true` on success, `false` otherwise.

5.78.2.2 void [params_free](#) (struct [params](#) * [params](#))

Frees a parameter structure.

Parameters

<i>params</i>	A pointer to the structure to free.
---------------	-------------------------------------

5.78.2.3 struct params* params_init (void) [read]

Initializes a parameters structure.

Returns

An initialized parameters structure.

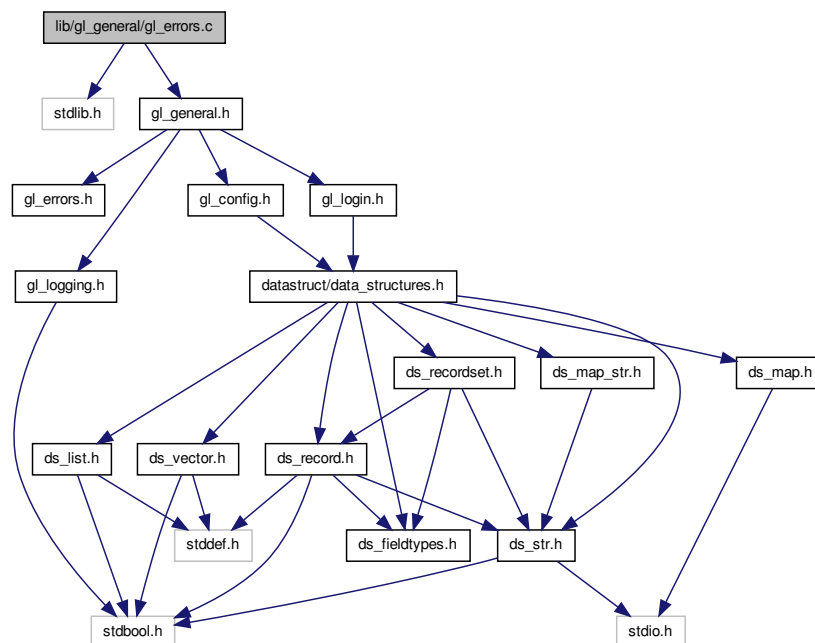
5.79 lib/gl_general/gl_errors.c File Reference

Implementation of error functionality.

```
#include <stdlib.h>
```

```
#include "gl_general.h"
```

Include dependency graph for gl_errors.c:



Functions

- void [gl_error_quit](#) (const char *msg)
Logs an error message and quits program.

5.79.1 Detailed Description

Implementation of error functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.79.2 Function Documentation**5.79.2.1 void gl_error_quit (const char * msg)**

Logs an error message and quits program.

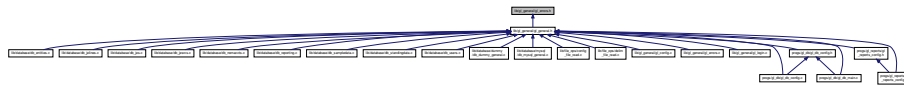
Parameters

<i>msg</i>	The error message to log.
------------	---------------------------

5.80 lib/gl_general/gl_errors.h File Reference

Interface to error functionality.

This graph shows which files directly or indirectly include this file:

**Functions**

- void [gl_error_quit](#) (const char *msg)
Logs an error message and quits program.

5.80.1 Detailed Description

Interface to error functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.80.2 Function Documentation**5.80.2.1 void gl_error_quit (const char * msg)**

Logs an error message and quits program.

Parameters

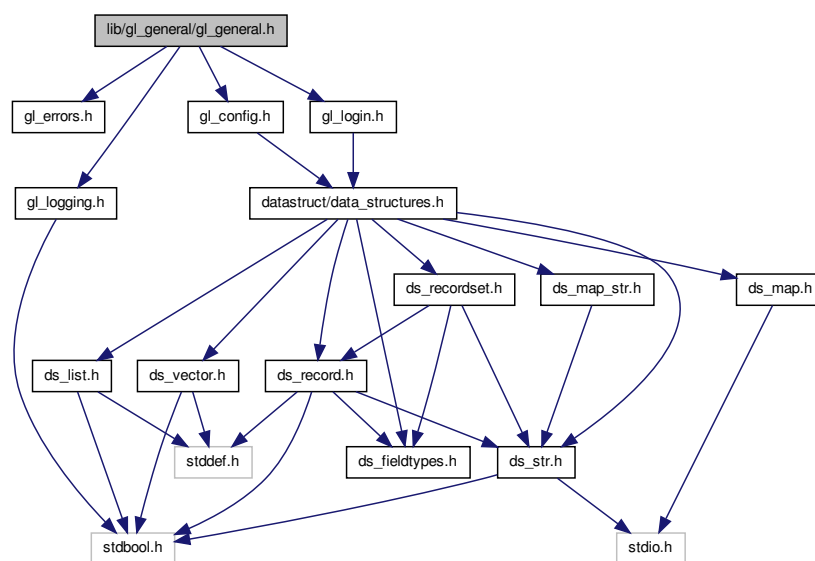
<i>msg</i>	The error message to log.
------------	---------------------------

5.81 lib/gl_general/gl_general.h File Reference

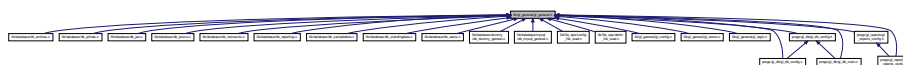
User interface to logging and error functionality.

```
#include "gl_errors.h"
#include "gl_logging.h"
#include "gl_login.h"
#include "gl_config.h"
```

Include dependency graph for gl_general.h:



This graph shows which files directly or indirectly include this file:



5.81.1 Detailed Description

User interface to logging and error functionality.

Author

Paul Griffiths

Copyright

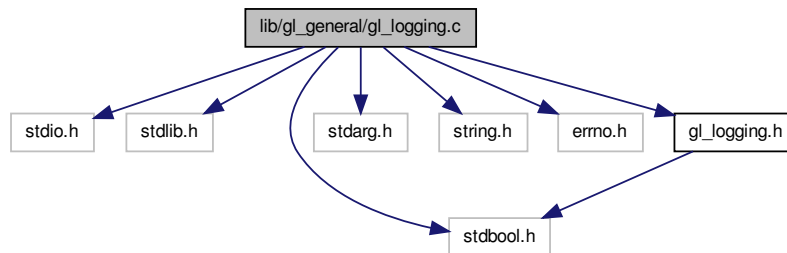
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.82 lib/gl_general/gl_logging.c File Reference

Implementation of logging functionality.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <stdarg.h>
#include <string.h>
#include <errno.h>
#include "gl_logging.h"
```

Include dependency graph for gl_logging.c:



Functions

- void [gl_set_logging](#) (const bool status)
Turns logging on or off.
- void [gl_log_msg](#) (const char *format,...)
Logs a message to the log file.

5.82.1 Detailed Description

Implementation of logging functionality. Implementation of logging functionality. Enables debugging and other system messages to be recorded to a log file.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.82.2 Function Documentation

5.82.2.1 void gl_log_msg (const char * format, ...)

Logs a message to the log file.

Logs a message to the log file.

Parameters

<i>format</i>	Format string, in same format as <code>printf()</code> .
...	Variable arguments as specified by format string.

5.82.2.2 void gl_set_logging (const bool status)

Turns logging on or off.

Turns logging on or off.

Parameters

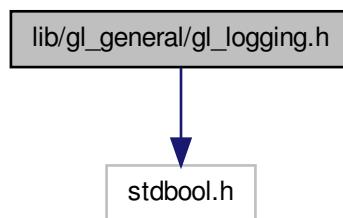
<i>status</i>	true to turn logging on, false to turn logging off.
---------------	---

5.83 lib/gl_general/gl_logging.h File Reference

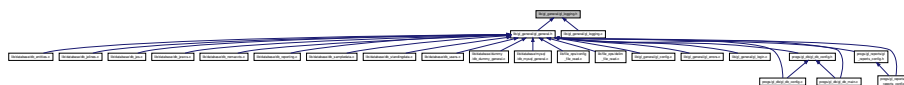
Interface to logging functionality.

```
#include <stdbool.h>
```

Include dependency graph for gl_logging.h:



This graph shows which files directly or indirectly include this file:



Functions

- void [gl_set_logging](#) (const bool status)
Turns logging on or off.
- void [gl_log_msg](#) (const char *format,...)
Logs a message to the log file.

5.83.1 Detailed Description

Interface to logging functionality. Interface to logging functionality. Enables debugging and other system messages to be recorded to a log file.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.83.2 Function Documentation**5.83.2.1 void gl_log_msg (const char * *format*, ...)**

Logs a message to the log file.

Logs a message to the log file.

Parameters

<i>format</i>	Format string, in same format as <code>printf()</code> .
...	Variable arguments as specified by format string.

5.83.2.2 void gl_set_logging (const bool *status*)

Turns logging on or off.

Turns logging on or off.

Parameters

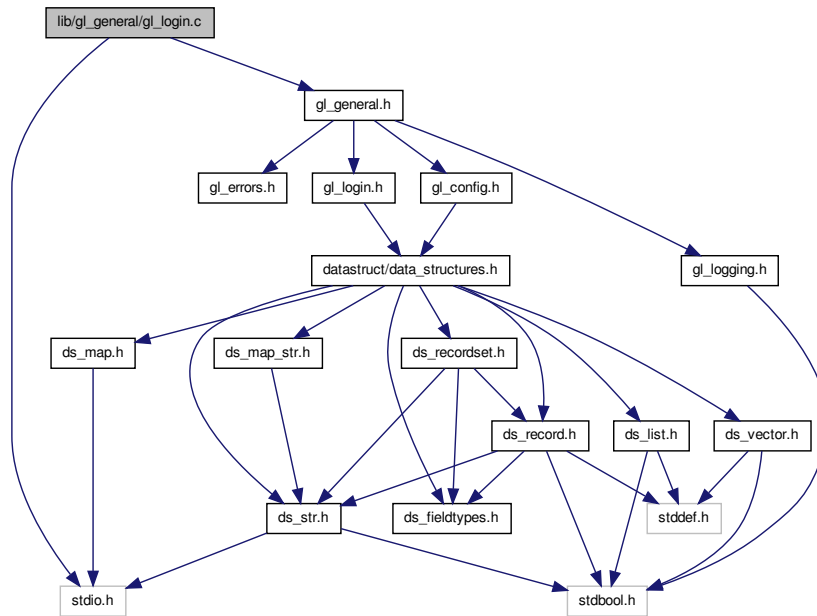
<i>status</i>	<code>true</code> to turn logging on, <code>false</code> to turn logging off.
---------------	---

5.84 lib/gl_general/gl_login.c File Reference

Implementation of login functionality.

```
#include <stdio.h>
#include "gl_general.h"
```

Include dependency graph for gl_login.c:



Functions

- [ds_str login](#) (void)
Gets a password from the user.

5.84.1 Detailed Description

Implementation of login functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.84.2 Function Documentation

5.84.2.1 ds_str login (void)

Gets a password from the user.

Returns

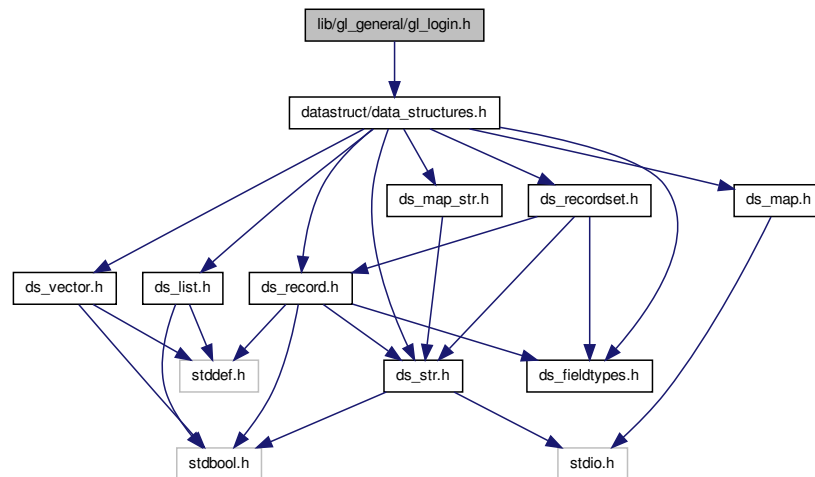
The password, or NULL on failure.

5.85 lib/gl_general/gl_login.h File Reference

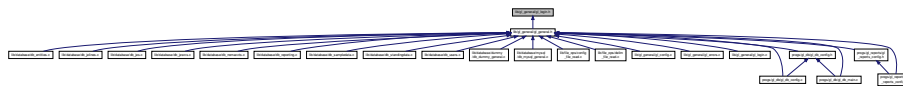
Interface to login functionality.

```
#include "datastruct/data_structures.h"
```

Include dependency graph for gl_login.h:



This graph shows which files directly or indirectly include this file:



Functions

- [ds_str login](#) (void)
Gets a password from the user.

5.85.1 Detailed Description

Interface to login functionality.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.85.2 Function Documentation

5.85.2.1 ds_str login (void)

Gets a password from the user.

Returns

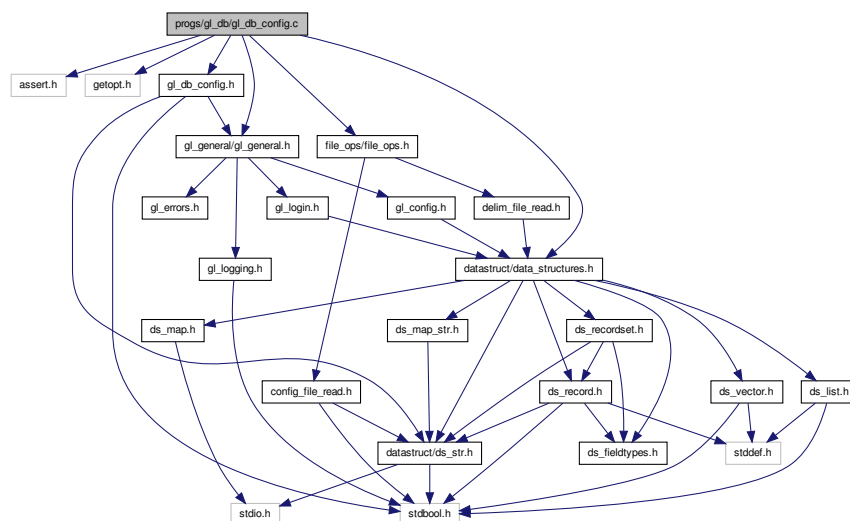
The password, or NULL on failure.

5.86 progs/gl_db/gl_db_config.c File Reference

Implementation of GL DB program configuration functionality.

```
#include <assert.h>
#include <getopt.h>
#include "gl_db_config.h"
#include "file_ops/file_ops.h"
#include "datastruct/data_structures.h"
#include "gl_general/gl_general.h"
```

Include dependency graph for gl_db_config.c:



Macros

- `#define _XOPEN_SOURCE 500`

Functions

- `bool get_cmdline_options (int argc, char **argv, struct params *params)`
Gets parameters from the command line.

5.86.1 Detailed Description

Implementation of GL DB program configuration functionality. Gets program configuration options from the command line and/or a configuration file.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.86.2 Macro Definition Documentation**5.86.2.1 #define _XOPEN_SOURCE 500**

UNIX feature test macro

5.86.3 Function Documentation**5.86.3.1 bool get_cmdline_options (int *argc*, char ** *argv*, struct params * *params*)**

Gets parameters from the command line.

Parameters

<i>argc</i>	<i>argc</i> as passed to main() .
<i>argv</i>	<i>argv</i> as passed to main() .
<i>params</i>	A pointer to a parameters structure to populate.

Returns

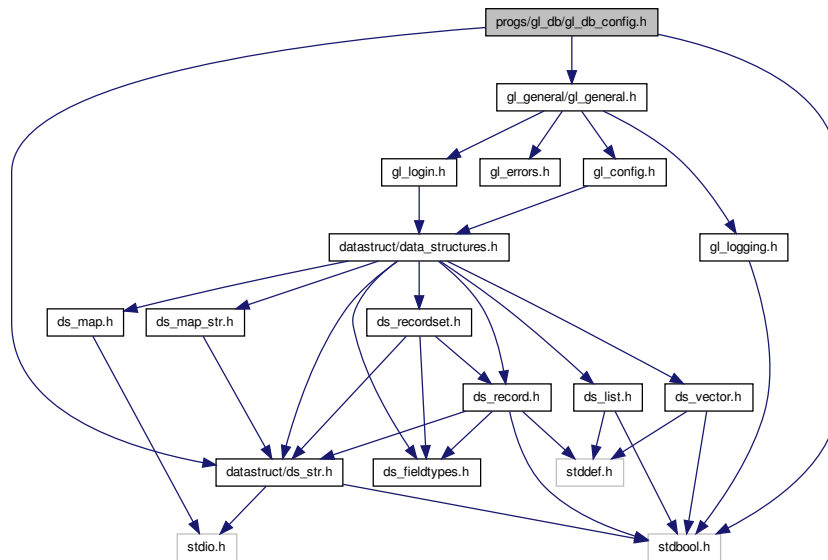
`false` if an unrecognized command line option was specified, `true` otherwise.

5.87 progs/gl_db/gl_db_config.h File Reference

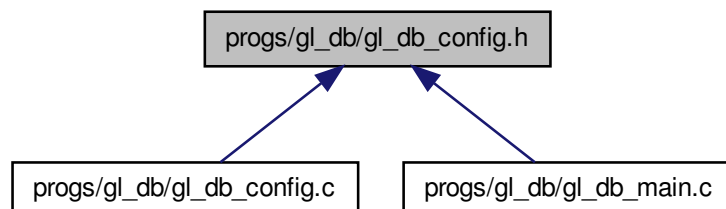
Interface to GL DB program configuration functionality.

```
#include <stdbool.h>
#include "datastruct/ds_str.h"
#include "gl_general/gl_general.h"
```

Include dependency graph for gl_db_config.h:



This graph shows which files directly or indirectly include this file:



Functions

- bool [get_cmdline_options](#) (int argc, char **argv, struct [params](#) *params)
Gets parameters from the command line.

5.87.1 Detailed Description

Interface to GL DB program configuration functionality. Gets program configuration options from the command line and/or a configuration file.

Author

Paul Griffiths

Prints a program help message.

- `int main (int argc, char **argv)`

Main function.

5.88.1 Detailed Description

Main function for GL database program. Main function for GL database program.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.88.2 Function Documentation

5.88.2.1 `int main (int argc, char ** argv)`

Main function.

Main function.

Returns

Exit status.

5.88.2.2 `void print_help_message (const char * progrname)`

Prints a program help message.

Parameters

<i>progrname</i>	The program name.
------------------	-------------------

5.88.2.3 `void print_usage_message (const char * progrname)`

Prints a program usage message.

Parameters

<i>progrname</i>	The program name.
------------------	-------------------

5.88.2.4 `void print_version_message (const char * progrname)`

Prints a program version message.

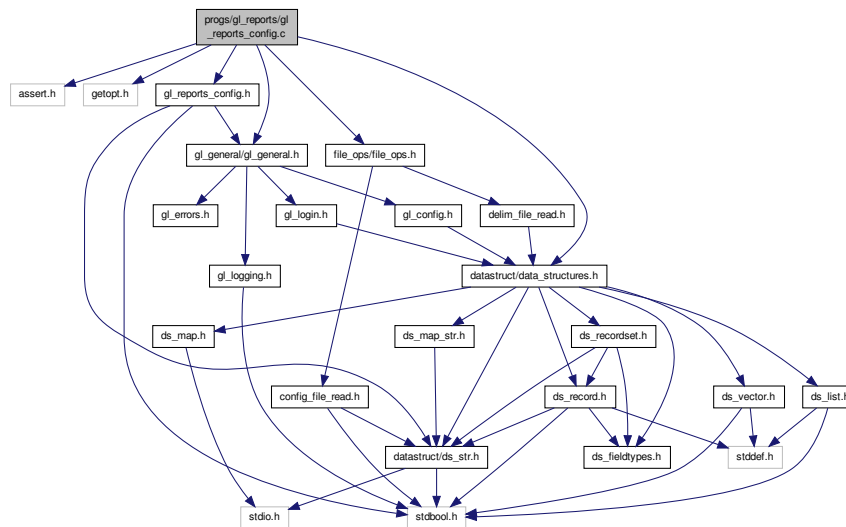
Parameters

<i>progrname</i>	The program name.
------------------	-------------------

5.89 progs/gl_reports/gl_reports_config.c File Reference

Implementation of GL reports program configuration functionality.

```
#include <assert.h>
#include <getopt.h>
#include "gl_reports_config.h"
#include "file_ops/file_ops.h"
#include "datastruct/data_structures.h"
#include "gl_general/gl_general.h"
Include dependency graph for gl_reports_config.c:
```



Macros

- `#define _XOPEN_SOURCE 500`

Functions

- `bool get_cmdline_options (int argc, char **argv, struct params *params)`
Gets parameters from the command line.

5.89.1 Detailed Description

Implementation of GL reports program configuration functionality. Gets program configuration options from the command line and/or a configuration file.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.89.2 Macro Definition Documentation

5.89.2.1 `#define _XOPEN_SOURCE 500`

UNIX feature test macro

5.89.3 Function Documentation

5.89.3.1 `bool get_cmdline_options (int argc, char ** argv, struct params * params)`

Gets parameters from the command line.

Parameters

<i>argc</i>	argc as passed to <code>main()</code> .
<i>argv</i>	argv as passed to <code>main()</code> .
<i>params</i>	A pointer to a parameters structure to populate.

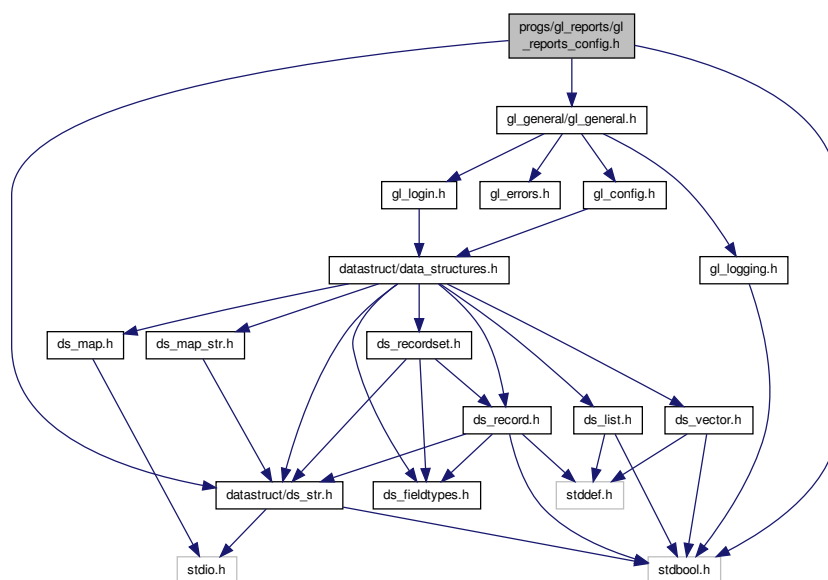
Returns

`false` if an unrecognized command line option was specified, `true` otherwise.

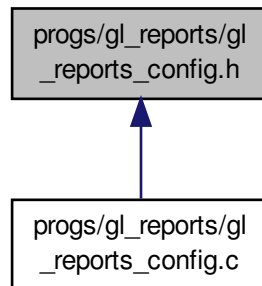
5.90 progs/gl_reports/gl_reports_config.h File Reference

Interface to GL reports program configuration functionality.

```
#include <stdbool.h>
#include "datastruct/ds_str.h"
#include "gl_general/gl_general.h"
Include dependency graph for gl_reports_config.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- bool `get_cmdline_options` (int argc, char **argv, struct `params` *params)
Gets parameters from the command line.

5.90.1 Detailed Description

Interface to GL reports program configuration functionality. Gets program configuration options from the command line and/or a configuration file.

Author

Paul Griffiths

Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

5.90.2 Function Documentation

5.90.2.1 bool `get_cmdline_options` (int *argc*, char ** *argv*, struct `params` * *params*)

Gets parameters from the command line.

Parameters

<i>argc</i>	argc as passed to <code>main()</code> .
<i>argv</i>	argv as passed to <code>main()</code> .
<i>params</i>	A pointer to a parameters structure to populate.

Returns

`false` if an unrecognized command line option was specified, `true` otherwise.

Index

- `_XOPEN_SOURCE`
 - `db_dummy_general.c`, 59
 - `gl_db_config.c`, 162
 - `gl_reports_config.c`, 167
- `CONFIG_FILE_OK`
 - `config_file_read.h`, 144
- `CONFIG_MAP_SIZE`
 - `config_file_read.c`, 141
- `capacity`
 - `ds_str`, 15
- `config_file_read`
 - `config_file_read.c`, 141
 - `config_file_read.h`, 144
- `config_file_read.c`
 - `CONFIG_MAP_SIZE`, 141
 - `config_file_read`, 141
 - `config_free`, 141
 - `config_init`, 141
 - `config_value_get`, 141
 - `config_value_get_cstr`, 142
 - `config_value_set`, 142
 - `MAX_BUFFER_SIZE`, 141
- `config_file_read.h`
 - `CONFIG_FILE_OK`, 144
 - `config_file_read`, 144
 - `config_free`, 144
 - `config_init`, 144
 - `config_value_get`, 145
 - `config_value_get_cstr`, 145
 - `config_value_set`, 145
- `config_free`
 - `config_file_read.c`, 141
 - `config_file_read.h`, 144
- `config_init`
 - `config_file_read.c`, 141
 - `config_file_read.h`, 144
- `config_value_get`
 - `config_file_read.c`, 141
 - `config_file_read.h`, 145
- `config_value_get_cstr`
 - `config_file_read.c`, 142
 - `config_file_read.h`, 145
- `config_value_set`
 - `config_file_read.c`, 142
 - `config_file_read.h`, 145
- `conn_mss`
 - `db_mysql_general.c`, 72
- `current`
 - `ds_list`, 9
- `ds_vector`, 16
- `DS_FIELD_BOOLEAN`
 - `ds_fieldtypes.h`, 78
- `DS_FIELD_DOUBLE`
 - `ds_fieldtypes.h`, 78
- `DS_FIELD_INT`
 - `ds_fieldtypes.h`, 78
- `DS_FIELD_STRING`
 - `ds_fieldtypes.h`, 78
- `data`
 - `ds_list_element`, 11
 - `ds_str`, 15
 - `ds_vector`, 16
- `data_destructor`
 - `ds_list`, 10
 - `ds_vector`, 16
- `database`
 - `params`, 18
- `db_connect`
 - `db_connection.h`, 21
 - `db_dummy_general.c`, 59
 - `db_mysql_general.c`, 71
- `db_connection.h`
 - `db_connect`, 21
- `db_create_database_structure`
 - `db_structure.c`, 51
 - `db_structure.h`, 52
- `db_create_entities_table`
 - `db_entities.c`, 22
 - `db_entities.h`, 24
- `db_create_entities_table_sql`
 - `db_dummy_create_entities_table_sql.c`, 56
 - `db_mysql_create_entities_table_sql.c`, 61
 - `db_sql.h`, 44
- `db_create_jelines_table`
 - `db_jelines.c`, 26
 - `db_jelines.h`, 28
- `db_create_jelines_table_sql`
 - `db_mysql_create_jelines_table_sql.c`, 62
 - `db_sql.h`, 44
- `db_create_jes_table`
 - `db_jes.c`, 29
 - `db_jes.h`, 31
- `db_create_jes_table_sql`
 - `db_mysql_create_jes_table_sql.c`, 63
 - `db_sql.h`, 44
- `db_create_jesrcs_table`
 - `db_jesrcs.c`, 32
 - `db_jesrcs.h`, 34

- db_create_jesrcs_table_sql
 - db_mysql_create_jesrcs_table_sql.c, 63
 - db_sql.h, 44
- db_create_nomaccts_table
 - db_nomaccts.c, 35
 - db_nomaccts.h, 37
- db_create_nomaccts_table_sql
 - db_mysql_create_nomaccts_table_sql.c, 64
 - db_sql.h, 44
- db_create_recordset_from_query
 - db_dummy_general.c, 59
 - db_mysql_general.c, 71
 - db_reporting.h, 40
- db_create_report_from_query
 - db_reporting.c, 39
 - db_reporting.h, 40
- db_create_standingdata_table
 - db_standingdata.c, 48
 - db_standingdata.h, 50
- db_create_standingdata_table_sql
 - db_mysql_create_standingdata_table_sql.c, 64
 - db_sql.h, 44
- db_create_users_table
 - db_users.c, 53
 - db_users.h, 55
- db_create_users_table_sql
 - db_dummy_create_users_table_sql.c, 56
 - db_mysql_create_users_table_sql.c, 65
 - db_sql.h, 45
- db_current_trial_balance_report
 - db_reporting.c, 39
 - db_reporting.h, 40
- db_current_trial_balance_report_sql
 - db_mysql_current_trial_balance_report_sql.c, 66
 - db_sql.h, 45
- db_delete_database_structure
 - db_structure.c, 51
 - db_structure.h, 52
- db_drop_entities_table
 - db_entities.c, 22
 - db_entities.h, 24
- db_drop_entities_table_sql
 - db_dummy_drop_entities_table_sql.c, 57
 - db_mysql_drop_entities_table_sql.c, 66
 - db_sql.h, 45
- db_drop_jelines_table
 - db_jelines.c, 26
 - db_jelines.h, 28
- db_drop_jelines_table_sql
 - db_mysql_drop_jelines_table_sql.c, 67
 - db_sql.h, 45
- db_drop_jes_table
 - db_jes.c, 29
 - db_jes.h, 31
- db_drop_jes_table_sql
 - db_mysql_drop_jes_table_sql.c, 67
 - db_sql.h, 45
- db_drop_jesrcs_table
 - db_jesrcs.c, 32
 - db_jesrcs.h, 34
- db_drop_jesrcs_table_sql
 - db_mysql_drop_jesrcs_table_sql.c, 68
 - db_sql.h, 45
- db_drop_nomaccts_table
 - db_nomaccts.c, 35
 - db_nomaccts.h, 37
- db_drop_nomaccts_table_sql
 - db_mysql_drop_nomaccts_table_sql.c, 69
 - db_sql.h, 45
- db_drop_standingdata_table
 - db_standingdata.c, 48
 - db_standingdata.h, 50
- db_drop_standingdata_table_sql
 - db_mysql_drop_standingdata_table_sql.c, 69
 - db_sql.h, 46
- db_drop_users_table
 - db_users.c, 53
 - db_users.h, 55
- db_drop_users_table_sql
 - db_dummy_drop_users_table_sql.c, 58
 - db_mysql_drop_users_table_sql.c, 70
 - db_sql.h, 46
- db_dummy_create_entities_table_sql.c
 - db_create_entities_table_sql, 56
- db_dummy_create_users_table_sql.c
 - db_create_users_table_sql, 56
- db_dummy_drop_entities_table_sql.c
 - db_drop_entities_table_sql, 57
- db_dummy_drop_users_table_sql.c
 - db_drop_users_table_sql, 58
- db_dummy_general.c
 - _XOPEN_SOURCE, 59
 - db_connect, 59
 - db_create_recordset_from_query, 59
 - db_execute_query, 59
- db_dummy_list_entities_report_sql.c
 - db_list_entities_report_sql, 60
- db_dummy_list_users_report_sql.c
 - db_list_users_report_sql, 61
- db_entities.c
 - db_create_entities_table, 22
 - db_drop_entities_table, 22
 - db_list_entities_report, 22
- db_entities.h
 - db_create_entities_table, 24
 - db_drop_entities_table, 24
 - db_list_entities_report, 24
- db_execute_query
 - db_dummy_general.c, 59
 - db_mysql_general.c, 71
 - db_query.h, 38
- db_jelines.c
 - db_create_jelines_table, 26
 - db_drop_jelines_table, 26
 - db_list_jelines_report, 26
- db_jelines.h

- db_create_jelines_table, 28
- db_drop_jelines_table, 28
- db_list_jelines_report, 28
- db_jes.c
 - db_create_jes_table, 29
 - db_drop_jes_table, 29
 - db_list_jes_report, 29
- db_jes.h
 - db_create_jes_table, 31
 - db_drop_jes_table, 31
 - db_list_jes_report, 31
- db_jesrcs.c
 - db_create_jesrcs_table, 32
 - db_drop_jesrcs_table, 32
 - db_list_jesrcs_report, 32
- db_jesrcs.h
 - db_create_jesrcs_table, 34
 - db_drop_jesrcs_table, 34
 - db_list_jesrcs_report, 34
- db_list_entities_report
 - db_entities.c, 22
 - db_entities.h, 24
- db_list_entities_report_sql
 - db_dummy_list_entities_report_sql.c, 60
 - db_mysql_list_entities_report_sql.c, 72
 - db_sql.h, 46
- db_list_jelines_report
 - db_jelines.c, 26
 - db_jelines.h, 28
- db_list_jelines_report_sql
 - db_mysql_list_jelines_report_sql.c, 73
 - db_sql.h, 46
- db_list_jes_report
 - db_jes.c, 29
 - db_jes.h, 31
- db_list_jes_report_sql
 - db_mysql_list_jes_report_sql.c, 74
 - db_sql.h, 46
- db_list_jesrcs_report
 - db_jesrcs.c, 32
 - db_jesrcs.h, 34
- db_list_jesrcs_report_sql
 - db_mysql_list_jesrcs_report_sql.c, 74
 - db_sql.h, 46
- db_list_nomaccts_report
 - db_nomaccts.c, 35
 - db_nomaccts.h, 37
- db_list_nomaccts_report_sql
 - db_mysql_list_nomaccts_report_sql.c, 75
 - db_sql.h, 46
- db_list_users_report
 - db_users.c, 54
 - db_users.h, 55
- db_list_users_report_sql
 - db_dummy_list_users_report_sql.c, 61
 - db_mysql_list_users_report_sql.c, 75
 - db_sql.h, 47
- db_mysql_create_entities_table_sql.c
 - db_create_entities_table_sql, 61
- db_mysql_create_jelines_table_sql.c
 - db_create_jelines_table_sql, 62
- db_mysql_create_jes_table_sql.c
 - db_create_jes_table_sql, 63
- db_mysql_create_jesrcs_table_sql.c
 - db_create_jesrcs_table_sql, 63
- db_mysql_create_nomaccts_table_sql.c
 - db_create_nomaccts_table_sql, 64
- db_mysql_create_standingdata_table_sql.c
 - db_create_standingdata_table_sql, 64
- db_mysql_create_users_table_sql.c
 - db_create_users_table_sql, 65
- db_mysql_drop_entities_table_sql.c
 - db_drop_entities_table_sql, 66
- db_mysql_drop_jelines_table_sql.c
 - db_drop_jelines_table_sql, 67
- db_mysql_drop_jes_table_sql.c
 - db_drop_jes_table_sql, 67
- db_mysql_drop_jesrcs_table_sql.c
 - db_drop_jesrcs_table_sql, 68
- db_mysql_drop_nomaccts_table_sql.c
 - db_drop_nomaccts_table_sql, 69
- db_mysql_drop_standingdata_table_sql.c
 - db_drop_standingdata_table_sql, 69
- db_mysql_drop_users_table_sql.c
 - db_drop_users_table_sql, 70
- db_mysql_general.c
 - conn_mss, 72
 - db_connect, 71
 - db_create_recordset_from_query, 71
 - db_execute_query, 71
 - main_mss, 72
- db_mysql_list_entities_report_sql.c
 - db_list_entities_report_sql, 72
- db_mysql_list_jelines_report_sql.c
 - db_list_jelines_report_sql, 73
- db_mysql_list_jes_report_sql.c
 - db_list_jes_report_sql, 74
- db_mysql_list_jesrcs_report_sql.c
 - db_list_jesrcs_report_sql, 74
- db_mysql_list_nomaccts_report_sql.c
 - db_list_nomaccts_report_sql, 75
- db_mysql_list_users_report_sql.c
 - db_list_users_report_sql, 75
- db_mysql_show_standingdata_report_sql.c
 - db_show_standingdata_report_sql, 76
- db_nomaccts.c
 - db_create_nomaccts_table, 35
 - db_drop_nomaccts_table, 35
 - db_list_nomaccts_report, 35
- db_nomaccts.h
 - db_create_nomaccts_table, 37
 - db_drop_nomaccts_table, 37
 - db_list_nomaccts_report, 37
- db_query.h
 - db_execute_query, 38
- db_reporting.c

- db_create_report_from_query, 39
- db_current_trial_balance_report, 39
- db_reporting.h
 - db_create_recordset_from_query, 40
 - db_create_report_from_query, 40
 - db_current_trial_balance_report, 40
- db_show_standingdata_report
 - db_standingdata.c, 48
 - db_standingdata.h, 50
- db_show_standingdata_report_sql
 - db_mysql_show_standingdata_report_sql.c, 76
 - db_sql.h, 47
- db_sql.h
 - db_create_entities_table_sql, 44
 - db_create_jelines_table_sql, 44
 - db_create_jes_table_sql, 44
 - db_create_jesrcs_table_sql, 44
 - db_create_nomaccts_table_sql, 44
 - db_create_standingdata_table_sql, 44
 - db_create_users_table_sql, 45
 - db_current_trial_balance_report_sql, 45
 - db_drop_entities_table_sql, 45
 - db_drop_jelines_table_sql, 45
 - db_drop_jes_table_sql, 45
 - db_drop_jesrcs_table_sql, 45
 - db_drop_nomaccts_table_sql, 45
 - db_drop_standingdata_table_sql, 46
 - db_drop_users_table_sql, 46
 - db_list_entities_report_sql, 46
 - db_list_jelines_report_sql, 46
 - db_list_jes_report_sql, 46
 - db_list_jesrcs_report_sql, 46
 - db_list_nomaccts_report_sql, 46
 - db_list_users_report_sql, 47
 - db_show_standingdata_report_sql, 47
- db_standingdata.c
 - db_create_standingdata_table, 48
 - db_drop_standingdata_table, 48
 - db_show_standingdata_report, 48
- db_standingdata.h
 - db_create_standingdata_table, 50
 - db_drop_standingdata_table, 50
 - db_show_standingdata_report, 50
- db_structure.c
 - db_create_database_structure, 51
 - db_delete_database_structure, 51
- db_structure.h
 - db_create_database_structure, 52
 - db_delete_database_structure, 52
- db_users.c
 - db_create_users_table, 53
 - db_drop_users_table, 53
 - db_list_users_report, 54
- db_users.h
 - db_create_users_table, 55
 - db_drop_users_table, 55
 - db_list_users_report, 55
- delim_file_read
 - delim_file_read.c, 147
 - delim_file_read.h, 148
- delim_file_read.c
 - delim_file_read, 147
 - MAX_LINE_SIZE, 147
- delim_file_read.h
 - delim_file_read, 148
- ds_fieldtypes.h
 - DS_FIELD_BOOLEAN, 78
 - DS_FIELD_DOUBLE, 78
 - DS_FIELD_INT, 78
 - DS_FIELD_STRING, 78
- ds_field_types
 - ds_fieldtypes.h, 78
- ds_fieldtypes.h
 - ds_field_types, 78
- ds_list, 9
 - current, 9
 - data_destructor, 10
 - ds_list.h, 84
 - free_on_delete, 10
 - head, 10
 - length, 10
 - tail, 10
- ds_list.c
 - ds_list_append, 80
 - ds_list_create, 80
 - ds_list_destroy, 80
 - ds_list_destructor, 80
 - ds_list_element, 80
 - ds_list_get_next_data, 81
 - ds_list_get_prev_data, 81
 - ds_list_is_empty, 81
 - ds_list_length, 81
 - ds_list_remove_all, 82
 - ds_list_remove_tail, 82
 - ds_list_seek_end, 82
 - ds_list_seek_start, 82
- ds_list.h
 - ds_list, 84
 - ds_list_append, 84
 - ds_list_create, 84
 - ds_list_destroy, 85
 - ds_list_destructor, 85
 - ds_list_element, 85
 - ds_list_get_next_data, 85
 - ds_list_get_prev_data, 86
 - ds_list_is_empty, 86
 - ds_list_length, 86
 - ds_list_remove_all, 86
 - ds_list_remove_tail, 86
 - ds_list_seek_end, 87
 - ds_list_seek_start, 87
- ds_list_append
 - ds_list.c, 80
 - ds_list.h, 84
- ds_list_create
 - ds_list.c, 80

- ds_list.h, [84](#)
- ds_list_destroy
 - ds_list.c, [80](#)
 - ds_list.h, [85](#)
- ds_list_destructor
 - ds_list.c, [80](#)
 - ds_list.h, [85](#)
- ds_list_element, [10](#)
 - data, [11](#)
 - ds_list.c, [80](#)
 - ds_list.h, [85](#)
 - next, [11](#)
 - previous, [11](#)
- ds_list_get_next_data
 - ds_list.c, [81](#)
 - ds_list.h, [85](#)
- ds_list_get_prev_data
 - ds_list.c, [81](#)
 - ds_list.h, [86](#)
- ds_list_is_empty
 - ds_list.c, [81](#)
 - ds_list.h, [86](#)
- ds_list_length
 - ds_list.c, [81](#)
 - ds_list.h, [86](#)
- ds_list_remove_all
 - ds_list.c, [82](#)
 - ds_list.h, [86](#)
- ds_list_remove_tail
 - ds_list.c, [82](#)
 - ds_list.h, [86](#)
- ds_list_seek_end
 - ds_list.c, [82](#)
 - ds_list.h, [87](#)
- ds_list_seek_start
 - ds_list.c, [82](#)
 - ds_list.h, [87](#)
- ds_map, [11](#)
 - ds_map.h, [91](#)
 - hash_size, [12](#)
 - lists, [12](#)
- ds_map.c
 - ds_map_destroy, [88](#)
 - ds_map_get_value, [88](#)
 - ds_map_init, [89](#)
 - ds_map_insert, [89](#)
 - ds_map_print_all, [89](#)
- ds_map.h
 - ds_map, [91](#)
 - ds_map_destroy, [91](#)
 - ds_map_get_value, [91](#)
 - ds_map_init, [91](#)
 - ds_map_insert, [91](#)
 - ds_map_print_all, [91](#)
- ds_map_destroy
 - ds_map.c, [88](#)
 - ds_map.h, [91](#)
- ds_map_get_value
 - ds_map.c, [88](#)
 - ds_map.h, [91](#)
- ds_map_init
 - ds_map.c, [89](#)
 - ds_map.h, [91](#)
- ds_map_insert
 - ds_map.c, [89](#)
 - ds_map.h, [91](#)
- ds_map_print_all
 - ds_map.c, [89](#)
 - ds_map.h, [91](#)
- ds_map_str, [12](#)
 - ds_map_str.h, [95](#)
 - hash_size, [12](#)
 - lists, [13](#)
- ds_map_str.c
 - ds_map_str_destroy, [93](#)
 - ds_map_str_get_value, [93](#)
 - ds_map_str_init, [93](#)
 - ds_map_str_insert, [93](#)
- ds_map_str.h
 - ds_map_str, [95](#)
 - ds_map_str_destroy, [95](#)
 - ds_map_str_get_value, [95](#)
 - ds_map_str_init, [95](#)
 - ds_map_str_insert, [96](#)
- ds_map_str_destroy
 - ds_map_str.c, [93](#)
 - ds_map_str.h, [95](#)
- ds_map_str_get_value
 - ds_map_str.c, [93](#)
 - ds_map_str.h, [95](#)
- ds_map_str_init
 - ds_map_str.c, [93](#)
 - ds_map_str.h, [95](#)
- ds_map_str_insert
 - ds_map_str.c, [93](#)
 - ds_map_str.h, [96](#)
- ds_record, [13](#)
 - ds_record.h, [101](#)
 - fields, [13](#)
- ds_record.c
 - ds_record_clear, [97](#)
 - ds_record_create, [97](#)
 - ds_record_destroy, [98](#)
 - ds_record_destructor, [98](#)
 - ds_record_get_field, [98](#)
 - ds_record_get_next_data, [98](#)
 - ds_record_make_delim_string, [98](#)
 - ds_record_make_values_string, [99](#)
 - ds_record_seek_start, [99](#)
 - ds_record_set_field, [99](#)
 - ds_record_size, [99](#)
 - ds_record_tokenize, [100](#)
- ds_record.h
 - ds_record, [101](#)
 - ds_record_clear, [102](#)
 - ds_record_create, [102](#)

- ds_record_destroy, 102
- ds_record_destructor, 102
- ds_record_get_field, 102
- ds_record_get_next_data, 102
- ds_record_make_delim_string, 103
- ds_record_make_values_string, 103
- ds_record_seek_start, 103
- ds_record_set_field, 103
- ds_record_size, 104
- ds_record_tokenize, 104
- ds_record_clear
 - ds_record.c, 97
 - ds_record.h, 102
- ds_record_create
 - ds_record.c, 97
 - ds_record.h, 102
- ds_record_destroy
 - ds_record.c, 98
 - ds_record.h, 102
- ds_record_destructor
 - ds_record.c, 98
 - ds_record.h, 102
- ds_record_get_field
 - ds_record.c, 98
 - ds_record.h, 102
- ds_record_get_next_data
 - ds_record.c, 98
 - ds_record.h, 102
- ds_record_make_delim_string
 - ds_record.c, 98
 - ds_record.h, 103
- ds_record_make_values_string
 - ds_record.c, 99
 - ds_record.h, 103
- ds_record_seek_start
 - ds_record.c, 99
 - ds_record.h, 103
- ds_record_set_field
 - ds_record.c, 99
 - ds_record.h, 103
- ds_record_size
 - ds_record.c, 99
 - ds_record.h, 104
- ds_record_tokenize
 - ds_record.c, 100
 - ds_record.h, 104
- ds_recordset, 14
 - ds_recordset.h, 110
 - field_lengths, 14
 - headers, 14
 - num_fields, 14
 - records, 14
 - types, 15
- ds_recordset.c
 - ds_recordset_add_record, 106
 - ds_recordset_create, 106
 - ds_recordset_destroy, 106
 - ds_recordset_get_next_insert_query, 106
- ds_recordset_get_text_report, 107
- ds_recordset_next_record, 107
- ds_recordset_num_fields, 107
- ds_recordset_num_records, 107
- ds_recordset_seek_start, 108
- ds_recordset_set_headers, 108
- ds_recordset_set_type, 108
- ds_recordset.h
 - ds_recordset, 110
 - ds_recordset_add_record, 110
 - ds_recordset_create, 110
 - ds_recordset_destroy, 111
 - ds_recordset_get_next_insert_query, 111
 - ds_recordset_get_text_report, 111
 - ds_recordset_next_record, 111
 - ds_recordset_num_fields, 111
 - ds_recordset_num_records, 112
 - ds_recordset_seek_start, 112
 - ds_recordset_set_headers, 112
 - ds_recordset_set_type, 112
- ds_recordset_add_record
 - ds_recordset.c, 106
 - ds_recordset.h, 110
- ds_recordset_create
 - ds_recordset.c, 106
 - ds_recordset.h, 110
- ds_recordset_destroy
 - ds_recordset.c, 106
 - ds_recordset.h, 111
- ds_recordset_get_next_insert_query
 - ds_recordset.c, 106
 - ds_recordset.h, 111
- ds_recordset_get_text_report
 - ds_recordset.c, 107
 - ds_recordset.h, 111
- ds_recordset_next_record
 - ds_recordset.c, 107
 - ds_recordset.h, 111
- ds_recordset_num_fields
 - ds_recordset.c, 107
 - ds_recordset.h, 111
- ds_recordset_num_records
 - ds_recordset.c, 107
 - ds_recordset.h, 112
- ds_recordset_seek_start
 - ds_recordset.c, 108
 - ds_recordset.h, 112
- ds_recordset_set_headers
 - ds_recordset.c, 108
 - ds_recordset.h, 112
- ds_recordset_set_type
 - ds_recordset.c, 108
 - ds_recordset.h, 112
- ds_str, 15
 - capacity, 15
 - data, 15
 - ds_str.h, 125
 - length, 15

ds_str.c

[ds_str_assign](#), 115
[ds_str_assign_cstr](#), 115
[ds_str_char_at_index](#), 115
[ds_str_clear](#), 115
[ds_str_compare](#), 116
[ds_str_compare_cstr](#), 116
[ds_str_concat](#), 116
[ds_str_concat_cstr](#), 116
[ds_str_create](#), 117
[ds_str_create_direct](#), 117
[ds_str_create_sprintf](#), 117
[ds_str_cstr](#), 117
[ds_str_decorate](#), 118
[ds_str_destroy](#), 118
[ds_str_destructor](#), 118
[ds_str_doubleval](#), 118
[ds_str_dup](#), 119
[ds_str_getline](#), 119
[ds_str_hash](#), 119
[ds_str_intval](#), 119
[ds_str_is_alnum](#), 120
[ds_str_is_empty](#), 120
[ds_str_length](#), 120
[ds_str_size_to_fit](#), 120
[ds_str_split](#), 121
[ds_str_strchr](#), 121
[ds_str_substr_left](#), 121
[ds_str_substr_right](#), 121
[ds_str_trim](#), 122
[ds_str_trim_leading](#), 122
[ds_str_trim_trailing](#), 122
[ds_str_trunc](#), 122

ds_str.h

[ds_str](#), 125
[ds_str_assign](#), 125
[ds_str_assign_cstr](#), 125
[ds_str_char_at_index](#), 125
[ds_str_clear](#), 126
[ds_str_compare](#), 126
[ds_str_compare_cstr](#), 126
[ds_str_concat](#), 126
[ds_str_concat_cstr](#), 126
[ds_str_create](#), 127
[ds_str_create_direct](#), 127
[ds_str_create_sprintf](#), 127
[ds_str_cstr](#), 128
[ds_str_decorate](#), 128
[ds_str_destroy](#), 128
[ds_str_destructor](#), 128
[ds_str_doubleval](#), 128
[ds_str_dup](#), 129
[ds_str_getline](#), 129
[ds_str_hash](#), 129
[ds_str_intval](#), 129
[ds_str_is_alnum](#), 130
[ds_str_is_empty](#), 130
[ds_str_length](#), 130

[ds_str_size_to_fit](#), 130
[ds_str_split](#), 131
[ds_str_strchr](#), 131
[ds_str_substr_left](#), 131
[ds_str_substr_right](#), 131
[ds_str_trim](#), 132
[ds_str_trim_leading](#), 132
[ds_str_trim_trailing](#), 132
[ds_str_trunc](#), 132

ds_str_assign

[ds_str.c](#), 115
[ds_str.h](#), 125

ds_str_assign_cstr

[ds_str.c](#), 115
[ds_str.h](#), 125

ds_str_char_at_index

[ds_str.c](#), 115
[ds_str.h](#), 125

ds_str_clear

[ds_str.c](#), 115
[ds_str.h](#), 126

ds_str_compare

[ds_str.c](#), 116
[ds_str.h](#), 126

ds_str_compare_cstr

[ds_str.c](#), 116
[ds_str.h](#), 126

ds_str_concat

[ds_str.c](#), 116
[ds_str.h](#), 126

ds_str_concat_cstr

[ds_str.c](#), 116
[ds_str.h](#), 126

ds_str_create

[ds_str.c](#), 117
[ds_str.h](#), 127

ds_str_create_direct

[ds_str.c](#), 117
[ds_str.h](#), 127

ds_str_create_sprintf

[ds_str.c](#), 117
[ds_str.h](#), 127

ds_str_cstr

[ds_str.c](#), 117
[ds_str.h](#), 128

ds_str_decorate

[ds_str.c](#), 118
[ds_str.h](#), 128

ds_str_destroy

[ds_str.c](#), 118
[ds_str.h](#), 128

ds_str_destructor

[ds_str.c](#), 118
[ds_str.h](#), 128

ds_str_doubleval

[ds_str.c](#), 118
[ds_str.h](#), 128

ds_str_dup

- ds_str.c, 119
 - ds_str.h, 129
- ds_str_getline
 - ds_str.c, 119
 - ds_str.h, 129
- ds_str_hash
 - ds_str.c, 119
 - ds_str.h, 129
- ds_str_intval
 - ds_str.c, 119
 - ds_str.h, 129
- ds_str_is_alnum
 - ds_str.c, 120
 - ds_str.h, 130
- ds_str_is_empty
 - ds_str.c, 120
 - ds_str.h, 130
- ds_str_length
 - ds_str.c, 120
 - ds_str.h, 130
- ds_str_size_to_fit
 - ds_str.c, 120
 - ds_str.h, 130
- ds_str_split
 - ds_str.c, 121
 - ds_str.h, 131
- ds_str_strchr
 - ds_str.c, 121
 - ds_str.h, 131
- ds_str_substr_left
 - ds_str.c, 121
 - ds_str.h, 131
- ds_str_substr_right
 - ds_str.c, 121
 - ds_str.h, 131
- ds_str_trim
 - ds_str.c, 122
 - ds_str.h, 132
- ds_str_trim_leading
 - ds_str.c, 122
 - ds_str.h, 132
- ds_str_trim_trailing
 - ds_str.c, 122
 - ds_str.h, 132
- ds_str_trunc
 - ds_str.c, 122
 - ds_str.h, 132
- ds_vector, 15
 - current, 16
 - data, 16
 - data_destructor, 16
 - ds_vector.h, 137
 - free_on_delete, 16
 - size, 16
- ds_vector.c
 - ds_vector_clear, 134
 - ds_vector_create, 134
 - ds_vector_destroy, 134
- ds_vector_destructor, 134
- ds_vector_element, 135
- ds_vector_get_next_data, 135
- ds_vector_seek_start, 135
- ds_vector_set, 135
- ds_vector_size, 135
- ds_vector.h
 - ds_vector, 137
 - ds_vector_clear, 137
 - ds_vector_create, 137
 - ds_vector_destroy, 138
 - ds_vector_destructor, 138
 - ds_vector_element, 138
 - ds_vector_get_next_data, 138
 - ds_vector_seek_start, 139
 - ds_vector_set, 139
 - ds_vector_size, 139
- ds_vector_clear
 - ds_vector.c, 134
 - ds_vector.h, 137
- ds_vector_create
 - ds_vector.c, 134
 - ds_vector.h, 137
- ds_vector_destroy
 - ds_vector.c, 134
 - ds_vector.h, 138
- ds_vector_destructor
 - ds_vector.c, 134
 - ds_vector.h, 138
- ds_vector_element
 - ds_vector.c, 135
 - ds_vector.h, 138
- ds_vector_get_next_data
 - ds_vector.c, 135
 - ds_vector.h, 138
- ds_vector_seek_start
 - ds_vector.c, 135
 - ds_vector.h, 139
- ds_vector_set
 - ds_vector.c, 135
 - ds_vector.h, 139
- ds_vector_size
 - ds_vector.c, 135
 - ds_vector.h, 139
- field_lengths
 - ds_recordset, 14
- fields
 - ds_record, 13
- free_on_delete
 - ds_list, 10
 - ds_vector, 16
- get_cmdline_options
 - gl_db_config.c, 162
 - gl_db_config.h, 164
 - gl_reports_config.c, 167
 - gl_reports_config.h, 168
- get_configuration

- gl_config.c, 150
- gl_config.h, 152
- gl_config.c
 - get_configuration, 150
 - params_free, 151
 - params_init, 151
- gl_config.h
 - get_configuration, 152
 - params_free, 152
 - params_init, 153
- gl_db_config.c
 - _XOPEN_SOURCE, 162
 - get_cmdline_options, 162
- gl_db_config.h
 - get_cmdline_options, 164
- gl_db_main.c
 - main, 165
 - print_help_message, 165
 - print_usage_message, 165
 - print_version_message, 165
- gl_error_quit
 - gl_errors.c, 154
 - gl_errors.h, 154
- gl_errors.c
 - gl_error_quit, 154
- gl_errors.h
 - gl_error_quit, 154
- gl_log_msg
 - gl_logging.c, 156
 - gl_logging.h, 158
- gl_logging.c
 - gl_log_msg, 156
 - gl_set_logging, 157
- gl_logging.h
 - gl_log_msg, 158
 - gl_set_logging, 158
- gl_login.c
 - login, 159
- gl_login.h
 - login, 160
- gl_reports_config.c
 - _XOPEN_SOURCE, 167
 - get_cmdline_options, 167
- gl_reports_config.h
 - get_cmdline_options, 168
- gl_set_logging
 - gl_logging.c, 157
 - gl_logging.h, 158
- hash_size
 - ds_map, 12
 - ds_map_str, 12
- head
 - ds_list, 10
- headers
 - ds_recordset, 14
- hostname
 - params, 18
- key
 - kv_pair_node, 17
- kv_pair_node, 16
 - key, 17
 - next, 17
 - value, 17
- length
 - ds_list, 10
 - ds_str, 15
- lib/database/database.h, 19
- lib/database/db_connection.h, 20
- lib/database/db_entities.c, 21
- lib/database/db_entities.h, 22
- lib/database/db_internal.h, 24
- lib/database/db_jelines.c, 25
- lib/database/db_jelines.h, 26
- lib/database/db_jes.c, 28
- lib/database/db_jes.h, 29
- lib/database/db_jesrcs.c, 31
- lib/database/db_jesrcs.h, 32
- lib/database/db_nomaccts.c, 34
- lib/database/db_nomaccts.h, 35
- lib/database/db_query.h, 37
- lib/database/db_reporting.c, 38
- lib/database/db_reporting.h, 39
- lib/database/db_sampledata.c, 41
- lib/database/db_sampledata.h, 41
- lib/database/db_sql.h, 42
- lib/database/db_standingdata.c, 47
- lib/database/db_standingdata.h, 48
- lib/database/db_structure.c, 50
- lib/database/db_structure.h, 51
- lib/database/db_users.c, 52
- lib/database/db_users.h, 54
- lib/database/dummy/db_dummy_create_entities_table_ -
_sql.c, 55
- lib/database/dummy/db_dummy_create_users_table_ -
_sql.c, 56
- lib/database/dummy/db_dummy_drop_entities_table_ -
_sql.c, 57
- lib/database/dummy/db_dummy_drop_users_table_ -
_sql.c, 57
- lib/database/dummy/db_dummy_general.c, 58
- lib/database/dummy/db_dummy_list_entities_report_ -
_sql.c, 60
- lib/database/dummy/db_dummy_list_users_report_sql_ -
c, 60
- lib/database/mysql/db_mysql_create_entities_table_ -
_sql.c, 61
- lib/database/mysql/db_mysql_create_jelines_table_sql_ -
c, 62
- lib/database/mysql/db_mysql_create_jes_table_sql.c,
62
- lib/database/mysql/db_mysql_create_jesrcs_table_sql_ -
c, 63
- lib/database/mysql/db_mysql_create_nomaccts_table_ -
_sql.c, 63

- lib/database/mysql/db_mysql_create_standingdata_table_sql.c, [64](#)
- lib/database/mysql/db_mysql_create_users_table_sql.c, [65](#)
- lib/database/mysql/db_mysql_current_trial_balance_report_sql.c, [65](#)
- lib/database/mysql/db_mysql_drop_entities_table_sql.c, [66](#)
- lib/database/mysql/db_mysql_drop_jelines_table_sql.c, [66](#)
- lib/database/mysql/db_mysql_drop_jes_table_sql.c, [67](#)
- lib/database/mysql/db_mysql_drop_jesrcs_table_sql.c, [68](#)
- lib/database/mysql/db_mysql_drop_nomaccts_table_sql.c, [68](#)
- lib/database/mysql/db_mysql_drop_standingdata_table_sql.c, [69](#)
- lib/database/mysql/db_mysql_drop_users_table_sql.c, [69](#)
- lib/database/mysql/db_mysql_general.c, [70](#)
- lib/database/mysql/db_mysql_list_entities_report_sql.c, [72](#)
- lib/database/mysql/db_mysql_list_jelines_report_sql.c, [73](#)
- lib/database/mysql/db_mysql_list_jes_report_sql.c, [73](#)
- lib/database/mysql/db_mysql_list_jesrcs_report_sql.c, [74](#)
- lib/database/mysql/db_mysql_list_nomaccts_report_sql.c, [74](#)
- lib/database/mysql/db_mysql_list_users_report_sql.c, [75](#)
- lib/database/mysql/db_mysql_show_standingdata_report_sql.c, [76](#)
- lib/datastruct/data_structures.h, [76](#)
- lib/datastruct/ds_fieldtypes.h, [77](#)
- lib/datastruct/ds_list.c, [78](#)
- lib/datastruct/ds_list.h, [82](#)
- lib/datastruct/ds_map.c, [87](#)
- lib/datastruct/ds_map.h, [89](#)
- lib/datastruct/ds_map_str.c, [92](#)
- lib/datastruct/ds_map_str.h, [94](#)
- lib/datastruct/ds_record.c, [96](#)
- lib/datastruct/ds_record.h, [100](#)
- lib/datastruct/ds_recordset.c, [104](#)
- lib/datastruct/ds_recordset.h, [108](#)
- lib/datastruct/ds_str.c, [113](#)
- lib/datastruct/ds_str.h, [122](#)
- lib/datastruct/ds_vector.c, [132](#)
- lib/datastruct/ds_vector.h, [136](#)
- lib/file_ops/config_file_read.c, [139](#)
- lib/file_ops/config_file_read.h, [142](#)
- lib/file_ops/delim_file_read.c, [145](#)
- lib/file_ops/delim_file_read.h, [147](#)
- lib/file_ops/file_ops.h, [148](#)
- lib/gl_general/gl_config.c, [149](#)
- lib/gl_general/gl_config.h, [151](#)
- lib/gl_general/gl_errors.c, [153](#)
- lib/gl_general/gl_errors.h, [154](#)
- lib/gl_general/gl_general.h, [155](#)
- lib/gl_general/gl_logging.c, [156](#)
- lib/gl_general/gl_logging.h, [157](#)
- lib/gl_general/gl_login.c, [158](#)
- lib/gl_general/gl_login.h, [160](#)
- lists
 - ds_map, [12](#)
 - ds_map_str, [13](#)
- login
 - gl_login.c, [159](#)
 - gl_login.h, [160](#)
- MAX_BUFFER_SIZE
 - config_file_read.c, [141](#)
- MAX_LINE_SIZE
 - delim_file_read.c, [147](#)
- main
 - gl_db_main.c, [165](#)
- main_mss
 - db_mysql_general.c, [72](#)
- next
 - ds_list_element, [11](#)
 - kv_pair_node, [17](#)
- num_fields
 - ds_recordset, [14](#)
- params, [17](#)
 - database, [18](#)
 - hostname, [18](#)
 - password, [18](#)
 - username, [18](#)
- params_free
 - gl_config.c, [151](#)
 - gl_config.h, [152](#)
- params_init
 - gl_config.c, [151](#)
 - gl_config.h, [153](#)
- password
 - params, [18](#)
- previous
 - ds_list_element, [11](#)
- print_help_message
 - gl_db_main.c, [165](#)
- print_usage_message
 - gl_db_main.c, [165](#)
- print_version_message
 - gl_db_main.c, [165](#)
- progs/gl_db/gl_db_config.c, [161](#)
- progs/gl_db/gl_db_config.h, [162](#)
- progs/gl_db/gl_db_main.c, [164](#)
- progs/gl_reports/gl_reports_config.c, [166](#)
- progs/gl_reports/gl_reports_config.h, [167](#)
- records
 - ds_recordset, [14](#)
- size
 - ds_vector, [16](#)

tail
 ds_list, [10](#)
types
 ds_recordset, [15](#)

username
 params, [18](#)

value
 kv_pair_node, [17](#)